MASTER EN SCIENCES INFORMATIQUES

Analyse et conception d'un Sifteo 2.0 dans le but d'apprendre les notions de programmation

Delhaye, Céline

Award date:
2020

Awarding institution:
Universite de Namur

UNIVERSITÉ DE NAMUR

Analyse et conception d'un Sifteo 2.0 dans le but d'apprendre les notions de programmation

Céline DELHAYE

RUE GRANDGAGNAGE, 21 ● B-5000 NAMUR(BELGIUM)

Résumé

Ce sujet intervient dans un contexte où l'apprentissage de la programmation et du numérique en général est devenu un enjeu crucial pour les jeunes. Tout au long de son parcours scolaire, l'étudiant doit comprendre et apprendre des concepts de plus en plus abstraits. Chacun a sa propre façon d'assimiler la matière. Il est donc important de trouver de nouvelles façons d'enseigner les matières. Certains profils d'apprentissage ont besoin de toucher, sentir, manipuler. Les interfaces tangibles utilisateur répondent en proposant un apprentissage par le jeu. C'est pourquoi, nous souhaitons utiliser une de ces interfaces tangibles afin de répondre aux problématiques rencontrées par les enseignants d'informatique. Au départ, les Sifteo cubes semblaient être un bon outil pour développer des pédagogies liées à ce type de profils. Cependant, le produit n'était plus sur le marché. Que faire dans ce cas ? Nous avons cherché à transposer techniquement cet outil et à implémenter les fonctionnalités par les Sifteo cubes. Grâce à notre analyse et nos recherches, nous avons pu prouver qu'il est au début possible de transposer les fonctionnalités clés. Cependant, la recherche n'est qu'au début, il faut parfaire l'interface et beaucoup d'analyses et de projets doivent être réaliser pour parvenir à un prototype.

Remerciements

Je tiens particulièrement à remercier mon promoteur, Mr B. Dumas, professeur à la faculté d'informatique de l'Université de Namur pour son aide et ses conseils dans le cadre de ce travail.

Je remercie toutes les personnes ayant de près ou de loin contribué à l'aboutissement de ce mémoire.

Table des matières

Chapitre 1

Introduction

1.1 Contexte

Ce sujet intervient dans un contexte où l'apprentissage de la programmation et du numérique en général est devenu un enjeu crucial pour les jeunes. En effet, la transformation digitale est déjà bien engagée et implique beaucoup de changements dans les métiers de demain. Ceux-ci demandent des compétences techniques et informatiques beaucoup plus pointues. Pour illustrer ce propos, Luc de Brabandère cite, dans une interview [2] pour présenter son nouveau livre [36], le commentaire d'un gérant de garage : « Avant j'embauchais des mécaniciens qui n'avaient pas peur de l'informatique. Aujourd'hui, j'engage des informaticiens qui n'ont pas peur de se salir les mains. ».

Face à ce constat, nos formations s'adaptent et l'apprentissage de la programmation est devenu une nécessité pour tous. Cependant, les difficultés sont de taille notamment parce que beaucoup de notions de programmation sont relativement abstraites (nous reviendrons sur ces difficultés plus loin dans ce travail). C'est pourquoi, il est vraiment pertinent de rechercher de nouveaux moyens plus adaptés aux jeunes pour enseigner cette matière.

1.2 Question de recherche et méthodologie

Un travail préliminaire sur les Sifteo en général nous a mené à dégager la question de recherche suivante :

"Peut-on créer et utiliser des Sifteo cubes 2.0 dans le but d'apprendre et comprendre la programmation pour les étudiants de baccalauréat ? ".

En analysant la phrase, nous nous rendons compte de l'utilité de définir les différents thèmes abordés. Pour ce faire, nous présenterons tout d'abord les Sifteo cubes. Dans un second temps, nous nous intéresserons au public cible et aux différents profils d'apprentissage plus précisément liés à la notion de programmation. Enfin, nous nous attarderons sur les difficultés rencontrées pour "apprendre et comprendre la programmation".

Dans le chapitre trois, nous développerons l'analyse de l'existant. Afin de répondre à la question de recherche, nous commencerons par réaliser un état de l'art. Cela consiste à examiner la littérature afin de découvrir les différentes expériences et recherches menées sur les Sifteo cubes et leurs utilisations. Les articles ont été trouvés grâce à des mots-clés :

"Sifteo cubes, tangible" etc... Ensuite, une première sélection a été réalisée sur base des différents titres. Parallèlement à cela, un tableau Excel a permis de dégager un classement : par domaine, par type d'utilité (apprentissage), par public visé, par technologie etc. Ce tableau permet aussi de visualiser et de retrouver facilement les différents articles. Lors de ces lectures, nous avons cherché des réponses aux questions suivantes :

— Quelles sont les fonctionnalités des Sifteo cubes ?
— Pourquoi cette technologie semble-t-elle intéressante dans l'apprentissage ?
— Quel est le public visé ?
— Quelles sont les compétences acquises via ces technologies ?

Signalons aussi que lorsque deux articles traitent du même sujet, une synthèse est fournie. Sur base des différents articles, nous avons dégagé la liste des fonctionnalités proposées.

Afin de compléter ma recherche, la réalisation d'un graphe a permis de mieux visualiser les domaines d'activités dans lesquels il y a eu des expériences Sifteo. Les domaines qui émergent sont ceux de l'éducation (notamment l'apprentissage de la programmation) et de la rééducation. Par ailleurs, les Sifteo cubes sont aussi utilisés comme technologie de support pour d'autres applications tangibles.

Nous verrons par la suite que les Sifteo cubes n'existent plus. C'est pourquoi, nous comparerons leurs fonctionnalités et technologies à celles de produits tangibles similaires. L'idée sera donc d'étudier la possibilité de transférer les différentes fonctionnalités vers une autre technologie.

Une fois cette étude terminée, nous proposerons un prototype non fonctionnel et nous implémenterons certaines fonctionnalités.

Notre but final n'est donc pas de reproduire à l'identique le Sifteo mais de réutiliser ses fonctionnalités principales afin de permettre aux étudiants de première année de comprendre des concepts abstraits liés à l'informatique : plus particulièrement les notions de fonction.

Pour terminer ce mémoire, nous discuterons des difficultés rencontrées durant la réalisation et proposerons d'éventuelles améliorations pour des futurs travaux.

Chapitre 2

Présentation du projet

2.1 Introduction

Dans ce chapitre, nous allons préciser comme prévu la question de recherche en présentant d'une part, les Sifteo cubes et en rappelant quelques notions pédagogiques liées à l'apprentissage de l'informatique, d'autre part.

2.2 Sifteo cubes

Parmi les différentes Interfaces Homme Machine, nous retrouvons les interfaces tangibles utilisateur, communément appelées TUI. Ce sont des objets physiques qui réagissent avec leur environnement et permettent à l'utilisateur de communiquer avec un système informatique [3] [4]. L'exemple le plus courant est la souris. En effet, l'utilisateur peut interagir avec le système via cet objet palpable. Un des objectifs du développement des TUI est de rendre l'interaction avec le système informatique plus ludique et plus pratique : une information numérique est remplacée par un mouvement concret. Il existe différentes catégories d'interfaces tangibles dont les cubes tangibles.

Parmi ces cubes tangibles, nous retrouvons le Sifteo cube (Fig 2.1) [5] qui est le sujet de notre étude.

Créés en 2009, par David Merrill et Jeevan Kalanithi, les Sifteo cubes sont des interfaces utilisateur tangibles et graphiques. Ces petits ordinateurs sont issus de la recherche Siftable [6] [technologie prototype développée au MIT Media Lab] . La société Sifteo inc. a été rachetée en 2004 par la compagnie 3D Robotics [7] [1]. Cependant, le software du programme Sifteo existe toujours en open source sur Github [2]. Malheureusement, le produit n'est plus vendu sur le marché . Les Sifteo cubes sont principalement destinés aux enfants de six à douze ans.

Pour ce qui concerne les produits Sifteo eux-mêmes, ce sont des blocs en forme de cube qui possèdent un écran couleur de 1,5 pouces permettant d'afficher des informations. Les Sifteo cubes se connectent à un ordinateur qui ordonne leur comportement. Grâce à des capteurs NFC, ils sont capables d'interagir, de communiquer entre eux dès lors que l'un de leurs côtés est mis à proximité d'un autre. Des accéléromètres à trois axes détectent tous les mouvements effectués. En résumé, chaque cube possède :

- un accéléromètre à 3 axes ;
- un écran LCD tactile ;
- une connexion sans fil et radio sans fil 2.4 Ghz ;
- une communication voisine : capteurs NFC ;

— un processeur ARM Cortex M3 , utilisé pour le traitement ;
— une batterie à piles rechargeable.

Le contact entre les blocs est détecté via une technologie NFC sans fil de très courte portée, ce qui nécessite très peu d'énergie et qui permet d'éviter une connexion câblée. Le cube possède sa propre batterie qui peut être rechargeable sur un socle ou par piles [8] .

Pour utiliser les Sifteo cubes, nous avons besoin d'un ordinateur ouvert durant toute la durée du jeu, la base Sifteo et les Sifteo cubes. La base permet d'activer ou de désactiver les cubes et de fournir les sons. C'est pourquoi les Sifteo doivent être assez proches de la base. Tous les jeux sont installés dans la mémoire du cube lancé à partir d'un ordinateur.

Dans le chapitre "Analyse de l'existant", nous parcourrons la littérature pour faire le point sur les expériences antérieures de ces cubes tangibles.

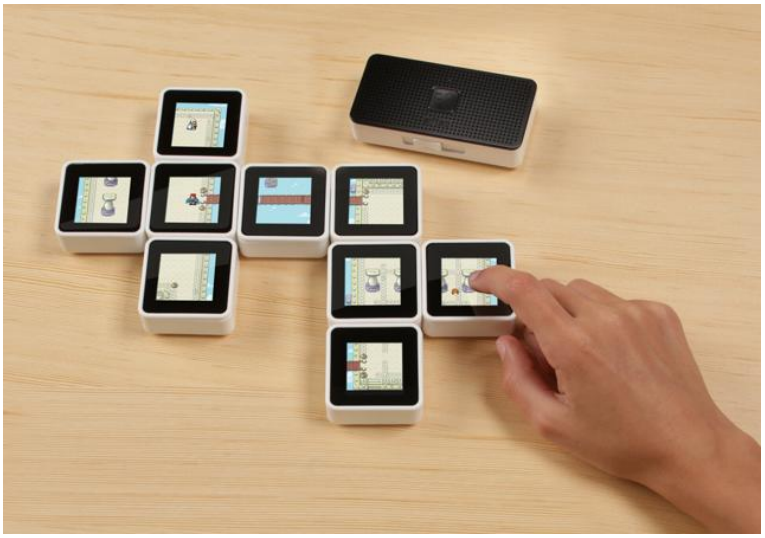

FIGURE 2.1 – Sifteo cubes et la base Sifteo

1. https ://3dr.com/
2. https ://github.com/sifteo/thundercracker

2.3 Aspects pédagogiques

Si nous voulons créer un nouveau produit, il est nécessaire de connaître le public cible. Afin de mieux comprendre celui-ci, nous présenterons dans cette section les différentes théories pédagogiques, les différentes notions informatiques que l'étudiant doit apprendre, les difficultés associées et nous proposerons d'éventuelles solutions.

2.3.1 Notions d'apprentissage selon Piaget

Le psychologue Jean Piaget [9] a défini quatre grandes phases de développement cognitif d'un enfant :

— le stade **sensorimoteur** (enfant de 0 à 2 ans) ;
— le stade **préopératoire** (2 à 7 ans) ;
— le stade **opérations concrètes** (7 à 11 ans) ;
— le stade **opérations formelles** (11 à approximativement 19 ans).

La plupart des recherches Sifteo cubes impliquant un jeu pour enfants travaille sur le deuxième et troisième stade.

Voici brièvement une explication de chaque stade.

Stade sensorimoteur : durant cette période, l'enfant appréhende le monde grâce à des expériences sensorielles.

Stade préopératoire : à ce stade, l'enfant est scolarisé. Il va devoir apprendre à être en contact avec d'autres et il va se poser beaucoup de questions.

Stade opérations concrètes : grâce aux différentes expériences du monde, les enfants savent conceptualiser et créer des raisonnements logiques dans certaines situations concrètes. Les notions abstraites sont encore trop complexes.

Stade opérations formelles : durant ce dernier stade, l'enfant a acquis les raisonnements logiques en incluant le raisonnement abstrait. Il est capable d'émettre des hypothèses sur des sujets qu'il ne maîtrise pas.

2.3.2 Profils d'apprentissage

Quelle que soit la méthode d'enseignement, chaque étudiant a sa propre motivation, sa propre façon de traiter et d'assimiler l'information. Dans les profils d'apprentissage [3], trois niveaux sont proposés. Nous nous concentrerons ici sur ce qui est essentiel pour notre objectif.

Tout d'abord les profils d'identité. Parmi ceux-ci, relevons :
— "le perfectionniste prend le temps de faire correctement les choses" ;
— "l'aimable travaille pour autrui" ;
— "le rebelle aime les défis ... et va jusqu'au bout de ses idées" ;
— "l'émotionnel réagit en fonction de ses émotions ... . Il a un côté créatif" ;
— "l'intellectuel a besoin de s'isoler pour réfléchir, il aime apprendre ..." ;
— "le dynamique est dans l'action. Il improvise et est fort débrouillard" ;

— "l'enthousiaste a la joie de vivre et a du mal à accepter l'autorité et les contraintes".

Parmi les profils de motivation, nous retrouvons ceux qui se posent ces questions :

— "Quelle utilité ? : motivation s'il trouve de l'intérêt au cours" ;
— "Vais-je apprendre ? : motivation par l'apprentissage, enrichir ses connaissances" ;
— "Avec qui ? : appréciation du cours en fonction de la personne qui l'enseigne" ;
— "Où ça se situe ? : il a besoin d'une vision globale et d'ordre dans l'apprentissage".

Parmi les profils de compréhension :

— le visuel : "visualiser à l'aide de couleurs, dessins, textes aérés avec une structure claire" ;
— l'auditif : "parler, écouter , se parler" ;
— le kinesthésique : "ressentir l'information, comprendre le pourquoi, saisir les origines, réaliser une application".

C'est dans le cadre kinesthésique que l'utilisation d'outils tangibles peut être très utile pour acquérir des notions par le concret et le mouvement.

2.3.3 Taxonomie de Bloom

Bloom et all. [10] ont développé une taxonomie [11] (Fig 2.2) pour les objectifs d'éducation à l'informatique. La taxonomie fournit une plateforme de six niveaux cognitifs à partir d'un niveau simple jusqu'à des niveaux plus complexes : connaissance, compréhension, application, analyse, synthèse et évaluation. Les trois premiers niveaux sont référencés comme étant peu complexes tandis que les trois derniers niveaux sont considérés comme des niveaux complexes et abstraits.

1. **la connaissance** : à ce stade, l'étudiant est capable de mémoriser les concepts ainsi que leur syntaxe ;
2. **la compréhension** : l'étudiant est capable d'interpréter un concept, de comprendre et d'expliquer la structure d'une méthode. Il doit appréhender le but de chaque structure utilisée dans le programme ;
3. **l'application** : l'étudiant applique et exécute une information apprise. L'étudiant utilise ses connaissances pour résoudre un problème ou les appliquer dans une nouvelle situation. En général, il s'agit d'un niveau complexe et le plus critique. L'étudiant novice en programmation passe rarement ce niveau.

Suite à une discussion avec des étudiants de première année, il semble que leur difficulté soit de mettre en pratique des notions théoriques et plutôt abstraites. Le passage d'un exemple pratique vu au cours à l'utilisation lors d'un travail pose souvent problème. C'est pour répondre à cette problématique que nous souhaitons utiliser les Sifteo cubes.

Parlons brièvement des trois niveaux supérieurs :

1. **l'analyse** : il s'agit de décomposer les informations et de trouver les relations entre elles ;
2. **la synthèse** : liée à l'écriture d'un nouveau programme ou à l'amélioration d'un existant en apportant des modifications. Selon la littérature, dans cette étape, les étudiants sont tenus de créer quelque chose sur base d'informations apprises ;
3. **l'évaluation** : demande un jugement face à un problème. Les étudiants doivent sélectionner la structure la plus efficace pour résoudre ce problème.

4. "https ://studentacademy.be/definis-ton-profil-dapprentissage"

Une personne performante dans les niveaux de complexité élevés est aussi performante dans les niveaux de complexité faibles. Ces six niveaux cognitifs sont souvent utilisés pour évaluer les compétences des étudiants en informatique.

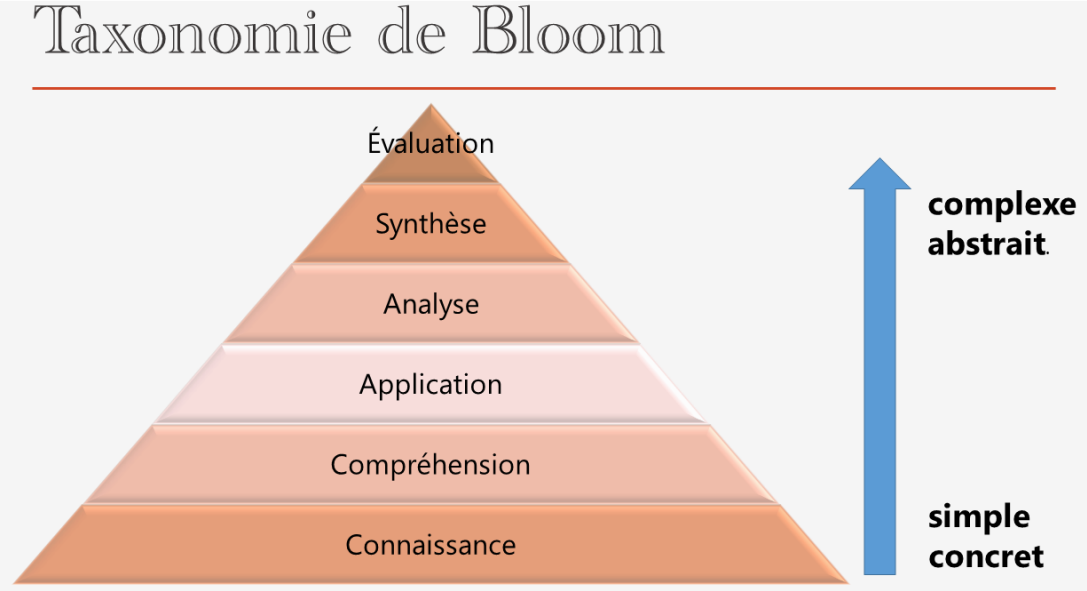

FIGURE 2.2 – Taxonomie de Bloom

Maintenant que nous connaissons les étapes d'apprentissage, passons en revue les notions à apprendre.

2.3.4 Notions de base de programmation

Apprendre les notions de programmation informatique est un réel challenge pour les étudiants qui sont au stade de l'initiation. Le taux d'échec en informatique en première année de baccalauréat est important. Les notions de base de la programmation ne sont pas chose aisée. C'est pourquoi nous cherchons un moyen de faciliter leur compréhension en utilisant de nouvelles méthodes d'apprentissage.

Rappelons les différentes notions [12] de base de la programmation :

— les séquences : c'est une suite d'instructions exécutées l'une après l'autre (séquentiellement) et qui, si elles sont correctement exécutées fournissent un résultat attendu. Ces instructions peuvent être des déclarations/affectations d'une variable, des opérations élémentaires sur une variable, l'affichage du contenu d'une variable ;

— les variables : dans un programme informatique, il est souvent utile de stoker de l'information au moins temporairement. C'est à ça que sert une variable. C'est un emplacement dans la mémoire identifié par un nom, une valeur (information stockée qui peut être modifié) et un type (numérique, alphanumérique ou autre) ;

— les structures conditionnelles : elles permettent de traiter des informations différemment en fonction d'une condition vraie ou fausse. Il existe différents types de structures conditionnelles : If(simple) - If()Else - switch(case) ;

— les structures itératives : cela permet de répéter des séquences d'instructions. Il existe deux types de boucles : for (où le nombre de répétitions est connu) et while (répétition tant que la condition est vérifiée) ;

— les fonctions : "une séquence d'instructions permettant de formaliser un concept (somme, IO)". Elles présentent beaucoup d'avantages : elles rendent le code plus lisible car elle permettent de le "moduler", elles servent aussi à éviter la redondance de code et, donc, d'avoir un point de modification unique. Le programme principale est appelé "fonction principale". Il existe plusieurs types de fonctions : les fonctions sans paramètre d'entrée et sans valeur de retour, les fonctions sans paramètre d'entrée et avec valeurs de retour, les fonctions avec paramètres d'entrées et sans valeur de retour, les fonctions avec paramètres d'entrées et avec valeurs de retour.

2.3.5 Difficultés

Sur base de la littérature, d'un sondage informel effectué auprès d'étudiants de première année et de mon expérience personnelle, il existe plusieurs raisons possibles à ces difficultés.

Tout d'abord, via un cours théorique, l'étudiant a du mal à se représenter le fonctionnement du code. En effet, il doit à la fois comprendre et apprendre la formalisation, tout en respectant une logique correcte et le "clean code". Cela revient à assimiler plusieurs notions complexes en parallèle.

Ensuite, pendant les séances pratiques, une autre compétence qui n'est pas facile est de comprendre et de se représenter un énoncé.

Par ailleurs, la motivation est un élément crucial car l'étudiant devra faire preuve de

persévérance dans son apprentissage. Lorsque le programme ne compile pas correctement, le programmeur doit montrer de la patience et avoir confiance.

Enfin, l'étudiant doit aussi pouvoir anticiper les solutions, c'est-à-dire s'imaginer les étapes de résolution pour arriver à la solution finale. Le transfert de connaissances qui lui est demandé pour passer des exemples de base aux exercices plus complexes est important et exige beaucoup d'entraînements et de "feedbacks".

2.3.6 Améliorations possibles

Pour enseigner des notions abstraites, il existe plusieurs méthodes. Dès le plus jeune âge, pour expliquer une notion abstraite, nous utilisons plusieurs exemples concrets afin que le cerveau de la personne puisse associer ces objets à quelque chose d'abstrait. Par exemple pour un enfant, la notion de rouge est abstraite, c'est pourquoi l'enseignant lui montre plusieurs objets concrets de couleur rouge afin qu'à un moment donné, l'élève comprenne que le point commun entre ces différents exemples est le rouge. Seul l'élève peut faire cette démarche intellectuelle.

Pour faire comprendre des notions abstraites aux étudiants de cycle supérieur, d'autres méthodes peuvent être utilisées. Une des méthodes consiste à motiver l'élève en donnant un intérêt à la notion, en expliquant en quoi cette notion lui sera utile . Enfin, une autre méthode clé est l'apprentissage par le jeu.

C'est pour ces raisons que les interfaces tangibles sont des outils très intéressants. Ils permettent aux étudiants d'apprendre à leur insu des notions de programmation.

Sur base des différents profils, des difficultés rencontrées et du type de jeux choisi, une petite analyse a été réalisée afin d'avoir un aperçu d'éventuelles solutions adaptées.

Sur l'image (Fig 2.3), sont représentés les types de jeux, les profils d'apprentissage que nous avons sélectionnés ainsi que les difficultés auxquelles les élèves sont confrontés. Les flèches représentent d'une part, les types de jeux adaptés aux profils et d'autre part, les types de jeux adaptés aux difficultés.
Nous pouvons donc constater (Fig 2.3) que les jeux sont des solutions bien adaptées pédagogiquement. Ils coïncident aussi aux aspirations des étudiants. N'oublions pas néanmoins que l'enseignement actuel convient parfaitement à certains profils.

Dans le chapitre suivant, nous proposerons l'état de l'art et nous établirons une comparaison des Sifteo par rapport à d'autres technologies.

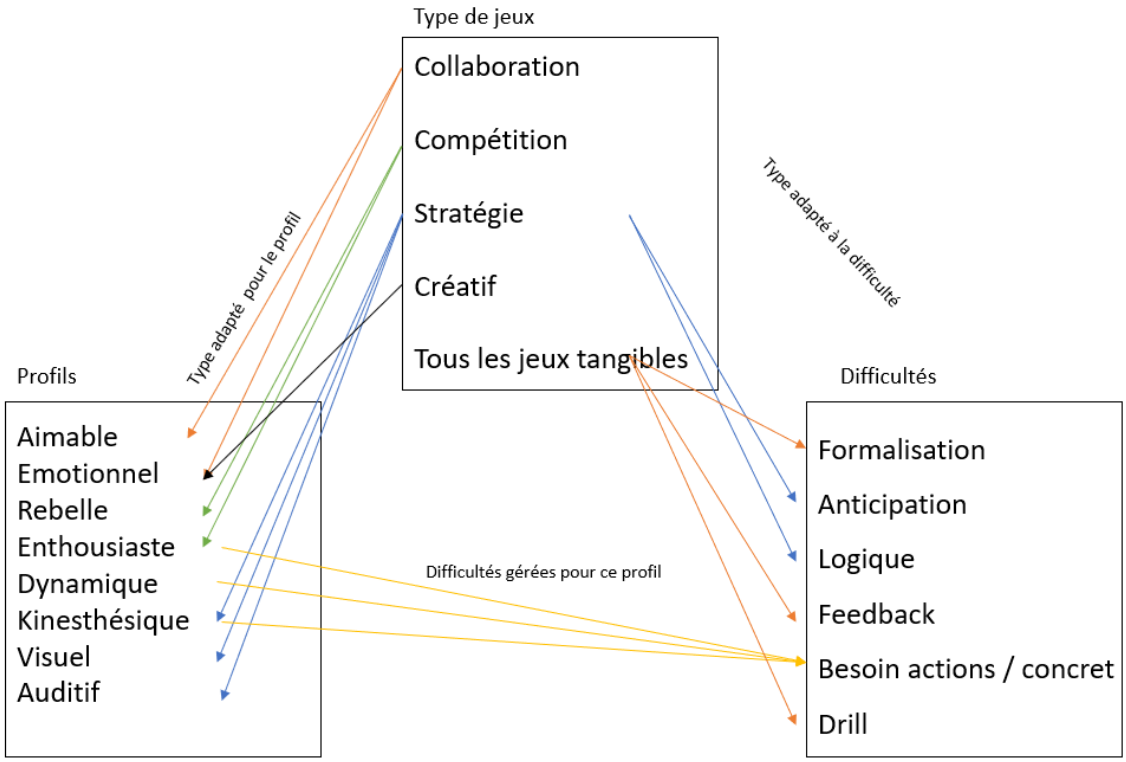

FIGURE 2.3 – Difficultés - jeux - solutions

Chapitre 3

Analyse de l'existant

3.1 Introduction

Dans ce chapitre, nous établissons l'état de l'art de l'utilisation du Sifteo comme outil pédadogique ou thérapeutique en étudiant différentes recherches reprises ci-dessous. Nous découvrirons dans ce chapitre, les domaines d'activités pour lesquels les Sifteo cubes sont utilisés, les différentes fonctionnalités qui ont été mises en oeuvre ainsi que les interactions auprès des utilisateurs. Enfin, nous comparerons Sifteo avec d'autres produits afin de mieux appréhender leurs limites fonctionnelles et technologiques.

3.2 Domaines d'activités

Nous allons présenter les quatre domaines d'activités suivants :
— Education chez les enfants
— Programmation
— Thérapies médicales
— Supports informatiques

3.2.1 Education chez les enfants

Les jeux éducatifs analysés dans le cadre de ce travail de fin d'études partagent tous les mêmes fonctionnalités de base (inclinaison, connexion, secouement, pression sur l'écran) mais organisent ces fonctionnalités de façons différentes.

Par ailleurs, les études mentionnées ci-dessous démontrent que Sifteo est un très bon outil pédagogique pour faciliter la formation des enfants agés de deux à treize ans. Par exemple, l'interface tangible permet d'acquérir tout en s'amusant les compétences suivantes : la communication interpersonnelle, les modes de collaboration en équipes, la représentation géo-spatiale, l'apprentissage grammatical et bien d'autres.

Exemple d'apprentissage plutôt centré sur le géo-spatial

le "Penguin Game" [13] consiste à aider un pingouin à ne pas se faire voler sa nourriture. Le "Fish Game" [13] accompagne un poisson dans sa quête de nourriture pour constituer une réserve. Ces deux jeux comportent plusieurs niveaux de difficultés et ont pour objectif d'apprendre les perspectives visuelles du monde et de permettre à l'enfant d'appréhender l'espace géométrique en suivant les différentes étapes proposées par le jeu.

L'analyse comportementale des enfants en train de jouer permet de constater les difficultés qu'ils éprouvent pour atteindre l'objectif proposé par le jeu. Un des problèmes constaté est celui de la latéralisation : les enfants différencient difficilement leur gauche et leur droite et ce, dès les premiers niveaux de difficultés, ce qui rend le jeu ardu à leurs yeux. Pour résoudre cette difficulté, les chercheurs ont adapté les jeux en proposant une aide supplémentaire aux enfants en coloriant la nageoire et l'aile droite des deux animaux. Les chercheurs ont aussi fait l'hypothèse que les enfants perçoivent mieux une situation en la regardant du ciel vers le sol. C'est pourquoi, ils ont conçus des objets facilement reconnaissables du dessus. Les chercheurs ont constaté que ces deux changements rendaient le jeu plus agréable, ce qui a permis d'étudier les stratégies d'apprentissage mises en oeuvre par les enfants. Ces derniers utilisent leur corps pour se représenter la situation géo-spatiale, ce qui tendrait à démontrer que les jeux offrant une représentation concrète de la réalité tout en permettant la méthode essai-erreur facilite et accélère le processus d'apprentissage du positionnement dans l'espace.

Exemple d'apprentissage plutôt centré sur la collaboration

Fat and Furious [1][14] est un jeu collaboratif impliquant plusieurs joueurs. Le jeu est composé de huit cubes. Sept cubes ont un écran représentant soit un chemin, soit un croisement. L'écran du huitième cube représente un hamster qui court à une vitesse croissante et choisit un des chemins au hasard. Le jeu s'arrête quand le hamster quitte un cube non connecté avec un autre ou rencontre un obstacle. Il est possible d'éliminer les obstacles en interagissant avec les cubes de la façon suivante : secouer, connecter deux cubes obstacles ensemble (l'eau et le feu), tapoter, retourner, pivoter etc... Ce jeu est destiné pour de courtes sessions permettant des interactions faciles et compréhensibles. Les concepteurs ont mis l'accent sur l'aspect collaboratif des joueurs. Le système a été pensé pour qu'une personne seule expérimentée soit moins efficace qu'un groupe ayant une expertise moyenne. En effet, la gestion anticipative et de plus en plus rapide des huit cubes requièrent plus de deux mains et une coordination entre les joueurs.

Exemple d'apprentissage de l'abstraction

Tangicons [15] est un jeu éducatif pour les enfants agés de six à neuf ans. Le but du jeu est de faire sortir la voiture ou le cheval d'un labyrinthe. Pour ce faire, les participants possèdent une série de cubes/jetons et un ordinateur. Chaque enfant possède son propre jeton. Les enfants jouent chacun à leur tour et ont donc le temps de réfléchir à la prochaine action qu'ils entreprendront. Le jeu ne peut pas se terminer si un des jetons est manquant. Cela favorise l'inclusion de tous les enfants : on gagne ou on perd ensemble. Contrairement à beaucoup de jeux qui stimulent la concurrence entre les enfants, ce jeu nécessite de la communication, de l'argumentation, de l'entraide entre les joueurs. Ils jouent vers un même but. Par ailleurs, il y a un décalage entre ce qui est représenté à l'écran de l'ordinateur et l'exécution des ordres par Sifteo. Ce décalage oblige les enfants à réfléchir sur l'impact de la décision qu'ils souhaitent prendre. Ceux-ci favorisent donc le sens de l'anticipation. De plus, les enfants ont besoin d'imaginer le positionnement des cubes (en d'autres termes de développer un degré d'abstraction) pour faire le lien entre le chemin indiqué par l'ordinateur et les icônes du cube. En outre, l'ordinateur indique la vue d'en haut, alors que les Sifteo cubes ne montrent que les directions gauche et droite et

1. https ://www.youtube.com/watch ?v=ff1isUfUW7E

les courbures. Cette dernière caractéristique induit également une capacité d'abstraction. Ce jeu est intéressant car l'anticipation et l'abstraction sont deux compétences visées par l'apprentissage de la programmation.
Une première évaluation a permis de démontrer que les enfants ont aimé le jeu et ont aimé jouer en équipe. Lors d'une deuxième évaluation, les cubes et l'ordinateur ont été placés sur le sol et la "station de charge" a été placée sur une table à huit mètres de distance. En plus de devoir sortir le cheval ou la voiture du labyrinthe, les enfants devaient veiller à ce que la voiture ou le cheval aient suffisamment d'énergie pour avancer. C'est la raison d'être du cube :" station de charge" qui vient d'être ajoutée. La deuxième évaluation a mis en évidence que les enfants concevaient des stratégies de jeu qui leur étaient propres. Certains avaient tendance à travailler avec la stratégie dite " essai-erreur", d'autres tournaient la tête afin de reconstruire la relation entre le virage sur la carte et le cube. Cet exercice stimule sans le savoir des processus cognitifs complexes qui se déroulent à des niveaux d'abstraction élevés. Le jeu permet de développer le raisonnement, la communication, la manipulation et la représentation conceptuelle. D'un point de vue informatique, les enfants apprennent la complexité liée à l'abstraction et les mécanismes de la programmation avec les notions de séquences, paramètres et les fonctions.

Exemple d'apprentissage du langage

"Make a Riddle" [16] cible les enfants agés de quatre à sept ans. Le but de ce jeu est d'apprendre les concepts de base permettant de construire une phrase. Les enfants peuvent construire des devinettes basées sur trois mots en organisant les cubes de telle façon qu'ils composent une expression. Cette expression sera représentée par un dessin sur un quatrième cube. Cela vise à développer la compétence du langage. Dans cet exercice, les enfants apprennent la phonétique et les mécanismes de base du langage.

"TéléStory" [16] cible les enfants de quatre à sept ans. Sur base d'une narration, ceux-ci participent à la création d'une scène animée sur une télévision. L'action à réaliser défile sur l'écran de télévision. Cela permet d'acquérir du vocabulaire et de s'initier à la lecture. Lors du lancement du jeu, des relations un à un ont été programmées entre les différents cubes. Si une de ces relations est fausse, l'écran émet une petite animation originale (ex : chat placé à côté d'une fleur sur le grand écran, le chat va renifler la fleur puis éternuer).

Une étude pilote a été réalisée sur des enfants interagissant avec "TéléStory" et "Make a Riddle". Il a été observé que les enfants comprennent facilement la connexion entre les cubes manipulés et la représentation graphique du contenu. Il a aussi été remarqué que les manipulations moins directes telles que l'inclinaison étaient moins évidente à comprendre. Par ailleurs, l'épisode narratif semblait efficace pour garder la concentration de l'enfant sur le jeu. Cependant, il y avait un conflit d'attention entre l'écran et la manipulation des cubes.
Avec "TéléStory", il est apparu que les enfants comprennent plus facilement les interactions avec les cubes grâce au descriptif narratif de la scène à représenter. Grâce à "Make a Riddle", les enfants ont un meilleur souvenir des phrases et ont acquis une meilleure appréhension des mécanismes liés à la conception de la phrase.

Exemple d'apprentissage centré sur la communication avec l'autre

"Maze Commander" [17] forme la capacité d'un joueur à communiquer correctement avec son partenaire. Ce jeu utilise deux types d'interactions différentes. Un des joueurs joue sur un Oculus Rift qui est un casque de réalité virtuelle tandis que l'autre joue avec des Sifteo cubes. Le but est de s'échapper d'un labyrinthe en évitant les ennemis et les dangers. La personne jouant avec Oculus Rift ne peut pas faire bouger le personnage mais possède une vue 3D du labyrinthe avec les ennemis et les cases piégées. La personne jouant avec les Sifteo cubes possède une vue constituée par quatre cubes (un cube correspond au personnage et les trois autres aux directions). Ce joueur doit déplacer le personnage dans le labyrinthe en manipulant les cubes grâce aux instructions données par le joueur de l'Oculus Rift. Le jeu requiert donc de la communication efficace et efficiente entre les deux joueurs.

Récapitulatif des fonctionnalités des Sifteo Cubes

le tableau récapitule les différentes fonctionnalités des Sifteo cubes utilisés dans les jeux :

- le contact entre cubes voisins
- le retournement
- l'inclinaison
- le secouement
- la pression écran
- l'affichage

Jeux et fonctionnalités Sifteo cubes						
Jeux	contact entre cubes voisin	Flip	Inclinaison	Secouement	Pression écran	Affichage
Pengiun Game	valide les positions (exécute l'instruction)	/	/	valide l'instruction (chasse l'obstacle)	donne l'instruction	(*)
Fish Game	valide les positions (va à l'endroit signalé)	/	/	/	donne l' instruction (indique où aller)	(*)
Fat & Furious	indique la route enlève les obstacles	/	enlève des obstacles	change de route	avance le personnage	(*)
Tangicons	définit un chemin	/	X	change de route	avance le personnage	(*)
TéléStory	valide la scène	change d'image	/	change d'image	/	(*)
Make a Riddle	forme une phrase	/	/	change de mot	/	(*)
Maze Commander	définit un chemin	X	X	change d'image	/	(*)

FIGURE 3.1 – Tableau récapitulatif des jeux et des fonctionnalités exploitées

Légende :
— / : intégré au jeu mais on ne sait pas pour quel aspect précis.
— X : intégré mais non utilisé.
— (*) : Affiche les messages du moment en fonction de la dernière instruction

3.2.2 Programmation

Les nouveaux dispositifs de type Sifteo cubes ont été utilisés aussi à des fins d'apprentissage de la programmation pour des étudiants de première année d'université. L'objectif est l'intérêt des jeunes vis-à-vis des cours de programmation et de leur permettre d'approfondir leur compréhension des concepts. Les résultats de quelques études montrent que les étudiants sont plus motivés à apprendre car Sifteo cubes permet de faire le lien entre le niveau concret et le niveau abstrait et d'introduire les notions de base de la programmation de cette façon.

Dans la première étude [18] qui a eu lieu dans une Université de Louisiane en 2012 et 2013, trois laboratoires proposent des dispositifs tangibles : l'un avec Arduino microcontrôleur qui est une plateforme hardware open source pour petits projets électroniques, le suivant avec Android phones avec AppInventor qui permet de développer des programmes pour Android mobile et, enfin, les Sifteo cubes. Ces derniers supports aident à l'apprentissage de la programmation événementielle car les étudiants voient la connexion entre les cubes et les objets qu'ils programment. Cela permet d'illustrer concrètement les concepts les plus abstraits du cours. Les chercheurs ont évalué l'efficacité des outils tangibles comme outil de motivation selon les deux façons suivantes :

- Tout d'abord, pour leur projet final, les étudiants avaient la possibilité de choisir la technologie à partir des choix suivants : Arduino project, AppInventor program, Sifteo Cube game, Python program (avec PyGame) ou Python program classique. Les chercheurs ont constaté sur une trentaine d'étudiants, plus de 75% d'entre eux ont choisi de réaliser un projet final avec des dispositifs tangibles : 40% ayant choisi Sifteo, 30% AppInventor et 7% Arduino.
- Ensuite, les chercheurs ont invité une dizaine d'étudiants à classer les laboratoires selon "l'utilité pour comprendre l'informatique". Une échelle de 1 à 5 où 1 était " inutile" à 5 "essentielle" a été utilisée pour évaluer les trois laboratoires. Le laboratoire Sifteo obtient une moyenne de 4,6 . Grâce à ces expériences, les étudiants se disent plus **créatifs**, plus motivés et se sentent mieux préparés à des projets de programmation événementielle plus complexes.

Une autre étude a montré l'impact des TUI dans la compréhension des concepts de programmation chez des étudiants. En effet, il n'est pas facile d'intégrer des concepts de programmation orientée objet. Un jeu basé sur les Sifteo cubes a été créé afin d'aider les étudiants à comprendre cette matière. Pour réaliser cette étude [19], deux groupes de quinze personnes ont appris les concepts différemment. Le premier groupe a travaillé classiquement c'est-à-dire que le professeur donne son cours oralement et les étudiants font des exercices sur PC. Le deuxième groupe a travaillé avec des Sifteo cubes. Afin d'enlever le facteur motivation, les élèves ont été sélectionnés parmi ceux qui avaient de grandes performances et montraient un intérêt au cours "Informatique fondamentale".

Pour évaluer la compréhension des concepts de base de la programmation orientée objet, les deux groupes ont été soumis à un test à choix multiples tant théorique que pratique. Les chercheurs ont retenu pour chaque étudiant la note globale obtenue au test, le temps mis pour réaliser le test, leur perception subjective de la difficulté et leur perception sur la clarté des exposés du professeur. Les résultats préliminaires montrent que les étudiants ayant travaillé avec des Sifteo cubes ont atteint un niveau d'apprentissage plus élevé. La technologie tangible a bien, selon les chercheurs, un impact positif sur l'apprentissage chez l'étudiant. Il est apparu dès lors qu'expliquer des concepts de base de l'orientée objet avec le soutien d'outils visibles et tangibles améliore et facilite l'apprentissage des étudiants.

Comparé au sous-point précédent, les fonctionnalités des Sifteo cubes ne sont pas mises en évidence, c'est pourquoi un tableau récapitulatif des fonctionnalités n'est pas possible.

3.2.3 Thérapies médicales

Les Sifteo cubes sont aussi utilisés dans le domaine médical à des fins thérapeutiques notamment pour les personnes cérébrolésées par des accidents vasculaires cérébraux [20], par la maladie d'Alzheimer [21] et bien d'autres. Plusieurs projets ont été développés à partir des Sifteo cubes pour rééduquer les patients. Nous en présentons quelques-uns ci-dessous.

Tout d'abord, le projet **REHAB@HOME** permet à des accidentés vasculaires cérébraux de se réhabituer à des mouvements simples principalement sur les membres supérieurs. Pour ce faire, cinq jeux de réhabilitation ont été déployés à travers des plateformes interactives (Kinect, Sifteo cubes, LeapMotion). Cela aide les patients à retrouver leur indépendance. Le jeu Sifteo cube "Simon Game" [20] avait pour but d'accorder deux couleurs. Lorsque l'utilisateur incline le Sifteo, il obtient une couleur ensuite il doit l'associer à un cube se trouvant devant lui. L'utilisateur réapprend dès lors à séquencer ses mouvements et à associer deux objets semblables.

Ensuite, le **Physicube** [22] est un prototype de jeux utilisé pour des personnes ayant un dysfonctionnement neurologique des membres supérieurs afin d'améliorer la mobilité de leurs bras. Le prototype Physicube est décomposé en deux types de jeux : LiftACube (Fig 3.2) et ReachAcube (Fig 3.3). Ils sont tous les deux basés sur la technologie Sifteo cubes. Le *LiftACube* permet de travailler les mouvements tels que lever, stabiliser, presser, retourner, secouer, connecter. L'un des jeux consiste à faire correspondre une clé et une serrure de la même couleur. Le patient possède six cubes dont cinq sont placés sur un plateau vertical avec des capteurs. Le dernier cube est libre et il possède un dessin d'une clé de couleur définie. Le dessin des serrures se trouve sur les cubes du plateau. L'utilisateur doit faire correspondre la clé et la serrure et stabiliser le cube pendant trois secondes. Lorsque le mouvement est bien réalisé, une animation détermine s'il y a concordance. Il est possible d'augmenter la difficulté en fonction du patient. Le *ReachACube* propose le même type de jeu mais sur un plan horizontal et se focalise ainsi sur l'entraînement des mouvements de latéralisation. Les Sifteo cubes doivent être placés sur la table à des positions différentes . Cependant, sans cadre de support, l'exercice devient difficile à cause des différents facteurs externes tels que la luminosité, l'inclinaison, le cadre renversé, l'éloignement de la zone de jeux etc. Pour éviter d'avoir ces difficultés, un cadre de support (Fig 3.3) a été réalisé pour l'exercice.

Les thérapeutes trouvent ces jeux intéressants et ont apprécié les prototypes. Ils disent que c'est une expérience concluante. Cependant, ils aimeraient que l'on démontre l'efficacité de ce type de thérapie sur le long terme.

Un autre jeu propose une cuisine tangible pour les personnes atteintes d'Alzheimer [21]. Ce système permet d'aider la personne à se remémorer et effectuer les opérations basiques réalisées quotidiennement en cuisine comme par exemple "se faire un café". Le patient possède une série de cubes afin de réaliser cette tâche. Le jeu permet de simuler les actions quotidiennes dans la cuisine en alliant le matériel et le son. Cela permet au patient de réapprendre par simulation. Les cubes montrent pas à pas les étapes à réaliser pour terminer correctement l'exercice.

FIGURE 3.2 – LiftACube

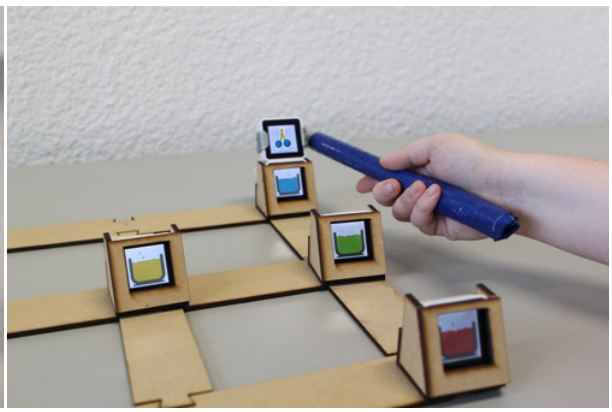

FIGURE 3.3 – ReachACube

Citons ensuite Cugcubed qui a développé un jeu grâce aux Sifteo cubes : "GroundsKeeper". Ce jeu utilise un set de quatre Sifteo cubes comme un outil d'évaluation des performances malgré l'augmentation des distractions. L'évaluation a lieu à partir du temps de réaction des utilisateurs. Le jeu dure plus ou moins 30 minutes avec seize niveaux. Grâce au Machine Learning, les ingénieurs ont développé un outil permettant de diagnostiquer des personnes atteintes du trouble de l'attention et de l'hyperactivité (TDAH). [23] [24]

Suite à cela, des chercheurs ont voulu d'abord examiner l'impact de la technologie Sifteo sur les enfants atteints du trouble d'hyperactivité avec déficit d'attention (TDAH) [25]. Il s'agit d'un trouble qui commence vers l'âge de 5 ans, les symptômes sont l'inattention, l'impulsivité et l'hyperactivité. Les chercheurs ont exploité l'utilisation d'objets kinesthésiques pour faire participer les enfants TDAH. Le jeu "WildFlower" a été développé par Frank Force en 2013. Il s'est basé, pour le choix du design et des sons, sur les recommandations de la littérature concernant les enfants TDAH. Le jeu a été testé avec cinq enfants (âgés de onze à dix-huit ans). Tous les participants ont joué l'entièreté de la session et ils n'ont pas été perturbés par des tests de distraction. Durant la session, des questions sur l'appréciation du jeu ont été posées, ils répondirent positivement sans interrompre l'engagement dans le jeu. Le jeu a duré une vingtaine de minutes et ils n'ont pas montré de signes d'hyperactivité, d'impulsivité ou d'inattention. Le deuxième jour, les utilisateurs ont d'abord joué dans une pièce pendant vingt minutes puis ont été invités à jouer à "WildFlower" dehors. Malgré le bruit environnant, ils montraient toujours un bon niveau d'engagement. La technologie a réussi à motiver, capter et tenir l'attention des enfants. Des études pourraient être menées dans un cadre plus professionnel et contrôlé.

Ci-dessous, un tableau récapitulatif des fonctionnalités des Sifteo cubes utilisées dans le cadre d'un jeu contrôlé permettant la rééducation d'une personne en déficit.

Fonctionnalités Sifteo cubes					
Jeux	Voisinage	Flip/Inclinaison	Secouement	Pression	Affichage
REHAB@HOME	valide l'instruction	donne la couleur	X	X	(*)
PhysiCube	valide l'instruction	change de clé	pour niveau supérieur	X	(*)
"Cuisine"	valide l'instruction	transfére entre objets	X	X	(*)

FIGURE 3.4 – Ttableau récapitulatif des fonctionnalités et leurs utilisations

Légende :
— X : intégrée mais non utilisée.
— (*) : Affiche les messages du moment en fonction de la dernière

3.2.4 Supports

Nous avons constaté que les Sifteo cubes étaient parfois utilisés comme support à d'autres technologies. Voici quelques exemples :

Tout d'abord, le **CubeQuery** [26] utilise conjointement les Sifteo cubes et un plateau interactif tactile, style Microsoft Surface SUR40. Cela permet de manipuler des requêtes de base de données. Chaque cube représente un paramètre de recherche individuel. Les cubes peuvent être placés côte à côte permettant de formuler des requêtes plus complexes. En les plaçant d'une certaine façon, les utilisateurs peuvent combiner des recherches en utilisant de simples booléens, des opérations AND ou des opérations OR. Deux cubes mis horizontalement appliquent une opération AND et verticalement une opération OR. Lorsque la requête est réalisée, les résultats sont affichés sur l'interface tactile avec un retour visuel et auditif. CubeQuery permet de rendre la recherche plus attractive et permet de réaliser des requêtes plus complexes simplement en alignant les dispositifs l'un à coté de l'autre.

Ensuite, dans cet article [27], les chercheurs souhaitent utiliser des widgets afin d'améliorer l'utilisation des interfaces tangibles. Ils ont donc proposé d'implémenter une série de widgets en utilisant les Sifteo cubes. Pour cela, plusieurs widgets ont été conceptualisés pour modifier un modèle BPMN2 (Business Process Model et Notion) sur une table interactive. Le Sifteo cubes doit posséder certaines caractéristiques propres pour interagir avec la table interactive, comme par exemple un écran montrant la nature du widget, un marqueur pour suivre le cube, un actionneur pour émuler la pression d'un bouton et des capteurs pour suivre l'orientation du cubes. Chaque widget possède sa propre identité afin de lui attribuer une fonctionnalité. Les chercheurs ont mis en place une table tangible où les cubes sont utilisés comme objets interactifs. Les Sifteo cubes deviennent des outils d'interaction tangibles.

Durant la session d'observation, un effet de bord a été observé : c'est l'aspect collaboratif. En effet, les participants distribuaient les travaux en fonction de l'usage d'un wigdet. Par exemple, l'un des participant appuyait sur le bouton tandis que l'autre déplaçait un autre objet tangible le long des éléments à supprimer.

Les chercheurs souhaitent utiliser des jetons (Sifteo cubes) actifs pour manipuler de grands ensembles de données. Dans cette recherche, les chercheurs ont voulu connaître les attentes des utilisateurs d'une interface tangible hybride et d'un langage gestuel. Une étude a été menée sur dix-neuf utilisateurs. Le but de l'expérience était de créer à partir de Sifteo cubes des gestes répondant à une série de sous-tâches à exécuter. Durant cette expérience, les utilisateurs devaient répondre à voix haute et valider leurs gestes. L'application ne répondait pas en temps réel et ne fournissait qu'un visuel de base, ce qui est l'inconvénient de l'expérience. Par exemple, si le cube était incliné, l'image se décalait pour simuler l'effet. Ensuite, les utilisateurs devaient donner une valeur de zéro à cinq pour exprimer la facilité du geste. Sans dialogue, ni repères entre l'utilisateur et le système, l'expérience n'était pas facile. Durant cette expérience, les utilisateurs étaient fort influencés par leurs propres expériences avec d'autres objets technologiques. Le travail a permis de fournir certaines contributions. La plupart des gestes étaient familiers et se déroulaient sur la surface et dans l'espace. Nous avons constaté une forte préférence pour les surfaces horizontales et verticales.

Dans cette section, le tableau des fonctionnalités récapitulatives n'est pas nécessaire. Elles varient en fonction de l'utilisation précise du cube.

3.3 Produits similaires

Dans cette section, nous mettrons en avant quelques produits similaires qui sont aussi des TUI susceptibles d'aider les enfants à comprendre et apprendre les notions de programmation. Voici donc la liste d'une sélection de quelques TUI repris :

— AlgoBlock
— E-Block
— Electronic Blocks
— ArBlocks
— AudioCube
— I-cubes.

Ci-dessous, nous présenterons un récapitulatif des produits similaires aux Sifteo cubes. Cela permettra de comparer les possibilités que présentent ceux-ci par rapport aux autres TUI de type cubiques.

AlgoBlock	
description [3]	le principe général est d'utiliser des cubes physiques et de les connecter les uns aux autres pour former une séquence d'actions qui sera interprétée par l'ordinateur afin de faire bouger quelque chose
technologie	inconnu

Comparaison avec les Sifteo cubes	
Différences	Ressemblances
les affichages se font sur l'ordinateur et non sur le cube	blocs connectés entre eux.

E-Block	
description [3]	il s'agit d'un jeu permettant de faire "sortir un avatar d'un labyrinthe grâce à des séquences programmées sur les cubes et les senseurs."
technologie [3]	les blocs possèdent une puce **microprocesseur**, un **transmetteur** et **récepteur** infrarouge afin de communiquer entre blocs, **module sans fil** pour communiquer avec l'ordinateur et un **module LED** permettant d'avertir des erreurs.

Comparaison avec les Sifteo cubes	
Différences	Ressemblances
transmetteurs et récepteurs infrarouges	module sans fil, microprocesseur
module LED pour les erreurs	affichage visuel des cubes
affichage non interactif	

Electronic Blocks	
description [3]	"blocs électroniques empilés les uns sur les autres pour former une structure pouvant interagir avec son environnement."
technologie [3]	"blocs en plastique composé d'un circuit électronique (placé à l'intérieur" du Lego)"

Comparaison avec les Sifteo cubes	
Différences	Ressemblances
pas d'affichage cubes peuvent être empilés	connexion via juste un côté

ArBlock [28] (Fig 3.5)	
description	il s'agit d'un jeu de réalité augmentée grâce auquel les informations sont projetées sur des blocs. Les blocs sont délimités par un cadre et identifiés via une trame particulière. L'espace utilisateur est limité à une zone.
technologie	caméra-projecteur

Comparaison avec les Sifteo cubes	
Différences	Ressemblances
pas d'écran intégré au cube projection et zone de jeu défini aucune technologie sur le cube	le design

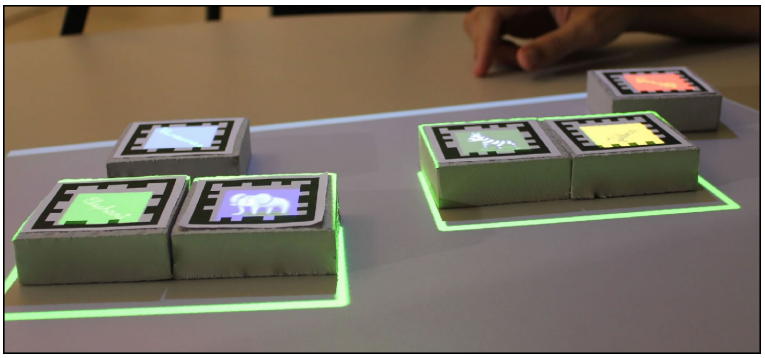

FIGURE 3.5 – Arblocks

AudioCube [29] (Fig 3.6)	
description	il s'agit d'un cube tangible possédant différentes LEDS permettant aux cubes d'interagir ensemble pour produire différents types de sons.
technologie	transmetteurs et récepteurs infrarouges avec des LEDS (rouge, bleu et vert), les senseurs reçoivent et envoient un signal audio, Audiocube possède sa propre batterie rechargeable.

Comparaison avec les Sifteo cubes	
AudioCubes	Sifteos
pas d'affichage juste des LEDS	
	batterie rechargeable
transmetteurs et émetteurs infrarouges	prototype NFC
communication 140/280Hz	communication sans fil radio 2.4 Ghz
sons émis par les cubes	sons émis par le socle
distance minimale pour être coupler à un autre cube 5 mm	2 mm

FIGURE 3.6 – AudioCubes

I-cubes [30]	
description	cube tangible qui permet aux enfants de comprendre et percevoir le monde. le cube est adapté pour les enfants de moins de 8 ans. le cube est employé pour deux types d'application : MusiCube (composition musicale tangible) et SpellingCube système interactif pour apprendre l'orthographe.
technologie	accéléromètre 3 axes , communication sans fil entre tête et cubes, tri-color LED, mini haut-parleur à vibration

Comparaison avec Sifteo cubes	
I-Cube	Sifteo
LEDS colors	feedback visuels (ou auditifs)
communication sans fil	communication sans fil
conscience spatiale 3D complète (6 faces)	juste les côtés de l'écran
détection orientation entre cubes	
4 récepteurs au milieu de chaque côté de la face	
écran LCD toutes les faces	écran LCD une face

3.4 Synthèse

Nous pouvons conclure que les Sifteo cubes, en tant qu'objets tangibles, peuvent aider les utilisateurs à développer l'anticipation, l'abstraction et la communication qui sont des éléments clés pour apprendre l'informatique. Tous ces mécanismes d'apprentissage peuvent être mis en oeuvre dans un projet pédagogique grâce aux fonctionnalités des Sifteo cubes. Selon la littérature, les Sifteo sont principalement utilisés dans des jeux d'apprentissage.

Les tableaux Fig 3.1 et Fig 3.4 présentent les différentes fonctionnalités des Sifteo cube présentes dans les jeux. Il permet d'identifier des cas d'utilisation en vue d'établir un nouveau jeu. Le tableau, ci-dessous, récapitule les jeux présentés dans la recherche pouvant répondre aux difficultés énoncées. Grâce à cette synthèse, nous pouvons établir un scénario global d'utilisation répondant à un maximum de critères.

Difficultés - Jeux adaptés										
Difficultés de l'étudiant	1	2	3	4	5	6	7	8	9	10
formalisation			v	v		v				
anticipation			v	v			v			v
logique	v	v	v	v	v	v	v	v	v	v
besoins d'actions			v							
suivis	v	v	v	v	v	v	v	v	v	v
entraînement	v	v	v	v	v	v	v	v	v	v

Légende :

- 1 - Penguin Game
- 2 - Fish Game
- 3 - Fat & Furious
- 4 - Tangicons
- 5 - Make a Riddle
- 6 - TéléStory
- 7 - Maze Commander
- 8 - Simon Game
- 9 - PhysiCube
- 10 - "Cuisine"

En analysant les différents cubes tangibles proches des Sifteo, nous pouvons réaliser un tableau récapitulatif des différentes fonctionnalités que proposent les Sifteo cubes par rapport à ses concurrents.

Fonctionnalités des Sifteo cubes	
Fonctionnalités utilisées	Fonctionnalités utiles non-utilisées
contact entre voisins	feedback auditif
animations sur écran	effet LED
effet inclinaison / flip	détection orientation cubes
effet secouement	empilement (sur l'écran)
pression écran tactile	conscience complète de toutes ces surfaces

Dans le chapitre qui suit, nous étudierons la possibilité de transposer les fonctionnalités désirées à d'autres technologies que le Sifteo cube.

Chapitre 4

Outils et fonctionnalités désirées

4.1 Introduction

Dans ce chapitre, dans un premier temps, nous décrivons les outils mis à notre disposition. Ensuite, nous émettons des hypothèses d'implémentation pour transposer les fonctionnalités avec d'autres technologies.

4.2 Problématique

Les Sifteo cubes n'étant plus sur le marché, nous souhaiterions savoir s'il est possible de recréer la structure du Sifteo cube sur base principalement d'une Raspberry Pi Zero. Pour l'affichage, plusieurs types d'écrans existent sur le marché, nous avons fait le choix de nous pencher sur le module ePaper PaPiRus. Visuellement, le ePaper connecté à une Raspberry ressemble à un rectangle tangible (Fig 4.3). Ce n'est pas le cas car d'une part, le Sifteo est une plateforme de jeu en connexion avec un ordinateur tandis que la Raspberry est un nano-ordinateur, d'autre part, la Raspberry ne possède pas actuellement les composants nécessaires pour simuler le Sifteo cube.

4.3 Outils utilisés

Dans cette section, nous allons détailler plus avant les principales technologies mises à notre disposition.

4.3.1 Raspberry Pi Zero

La Raspberry Pi Zero [1] [31] est un nano-ordinateur ne disposant pas de disque dur interne. Elle requiert pour fonctionner correctement, au moins un stockage mémoire de type carte SD selon le modèle et une source d'alimentation électronique. Cette source nécessite soit une pile, soit une batterie ou soit une alimentation micro USB.

Pour pallier au manque de disque dur interne, les données ont été stockées sur une carte SD. Celle-ci comprend le système d'exploitation linux open source. La Raspberry Pi Zero par sa petite taille permet de tester la création de petits projets, tels que la création d'un serveur web, d'une console rétro gaming, etc... La Raspberry pi Zero possède notamment :

— 1 GHz single coeur CPU
— 512 MB de RAM
— un port mini HDMI permettant de connecter un écran
— un port micro-USB

— un port alimentation micro-US OTG (On-The-Go)
— un HAT avec 40 pins
— un connecteur caméra CSI
— un module bluetooth
— un module Wifi

La raspberry pi Zero (Fig 4.1) communique avec le module ePaper PaPiRus (Fig 4.2) grâce à ses pins.

4.3.2 EPaper PaPiRus

"PaPiRus [1]- the ePaper Screen HAT for your raspberry Pi" [32] est un projet créatif mis en place sur la plateforme de financement participatif, KickStarter [2]. PaPiRus est une solution d'affichage compatible HAT permettant d'installer des écrans ePaper/eInk sur des Raspberry Pi. L'idée a été développée par Aaron Shaw qui a fondé l'entreprise PiSupply. Il s'agit d'un créateur d'accesoires innovants pour mini-ordinateur tel que "Raspberry". Voici une série de composants du PaPiRus :

— une conception conforme à la Raspberry PI HAT
— un écran de taille différente interchangeable
— une mémoire flash de 32 MBits
— une horloge à temps réel avec batterie
— des capteurs de température numérique
— connexion de dérivation GPIO,
— 5 boutons-poussoirs.
— convient pour une alimentation 3.3v ou 5V pour une utilisation RasPi, Arduino etc..

ePaper est une technologie qui permet d'imiter l'encre sur le papier. Il s'agit d'un écran à basse consommation pour mini-ordinateur. ePaper est capable de garder en mémoire des images et des textes sans électricité pendant une très longue période avant de s'effacer lentement. ePaper réfléchit la lumière, ce qui nous permet de lire sur l'écran à la lumière du jour sans aucun reflet. Citons encore quelques caractéristiques de ePaper :

— ultra basse consommation
— lisible au soleil
— haute résolution
— mince et facile à intégrer
— remplacement du papier
— détection de la casse.

1. https ://uk.pi-supply.com/products/papirus-epaper-eink-screen-hat-for-raspberry-pi
2. https ://www.kickstarter.com

Ci-dessous, un tableau récapitulatif des forces et des faiblesses du ePaper PaPiRus.

Forces	Faiblesses
adapté pour des applications avec peu de rafraîchissement	pas intéressant pour applications nécessitant un rafraîchissement rapide
adapté pour des applications avec usage faible consommation	peu de polices d'écriture
excellent angle de vision (180 °)	
résolution graphique avec monochrome	images monochromes
pas consommation pour l'affichage	

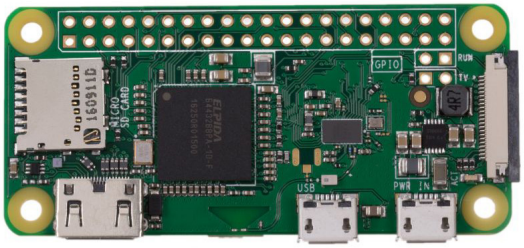

FIGURE 4.1 – Raspberry Pi Zero

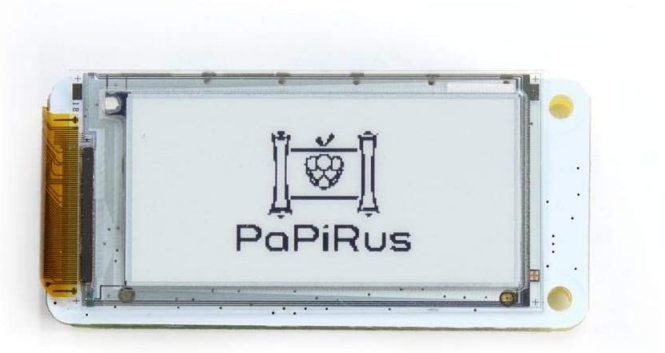

FIGURE 4.2 – ePaper Papirus

FIGURE 4.3 – Raspberry + ePaper PaPiRus

4.3.3 Capteurs

Les Sifteo cubes ont la particularité de détecter les différents mouvements de l'utilisateur. Pour ce faire, le cube possède un accéléromètre 3 axes. Cela permet au cube de réagir lorsqu'il est manipulé par un utilisateur. Pour recréer cet aspect-là, nous avons besoin de capteurs IMU [33], "Inertial measurement units". Ce sont des capteurs permettant de comprendre l'orientation d'un corps dans un espace tridimensionnel. Le produit utilisé est un capteur MPU-9250 [34] avant trois composants :

- un accéléromètre [35] : "Ils sont utilisés pour enregistrer à la fois les accélérations statiques (la gravité) et dynamiques (mouvement)."
- un gyroscope : "Il permet de calculer la vitesse de rotation autour d'un axe."
- un magnétomètre : "ils utilisent les champs magnétiques terrestre pour comprendre la direction."

« Une des applications principale des accéléromètres est le calcul d'inclinaison. Grâce à l'effet de la gravité, un accéléromètre peut vous dire comment est orienté votre objet par rapport à la terre. » [35]

L'accéléromètre calcule l'accélération linéaire dont l'unité est le m/s^2. Les valeurs générées par l'accéléromètre sont exprimées en g (environ 9,81 m/s^2) et elles varient entre [-2g ; 2g].

FIGURE 4.4 – Capteur MPU9250

4.4 Hypothèses de solutions

Dans cette section, nous envisagerons quelques hypothèses de solutions afin de recréer la structure d'un Sifteo cube. Voici la liste des fonctionnalités intéressantes du Sifteo cube :

— connexion avec cubes voisins afin de transmettre des informations ;
— inclinaison du cube (utilité en fonction du jeu) ;
— pression sur l'écran tactile (utilité en fonction du jeu) ;
— agitation du cube (utilité en fonction du jeu) ;
— affichage d'informations sous forme d'images ou de textes.

Il semble que la fonctionnalité principale consiste à pouvoir connecter des cubes entre eux afin de s'échanger des informations. Une autre façon d'interagir avec le cube est l'inclinaison et l'agitation. Chaque jeu exploite ces fonctionnalités différemment.

Ci-dessous, voici une synthèse des différentes fonctionnalités à transférer et pour chacune d'elles, une hypothèse de solution :

Fonctionnalités	Technologie Sifteo	PaPiRus/Raspberry Pi Zero
contact entre cubes voisins	technologie NFC	?
inclinaison	accéléromètre 3 axes	accéléromètre 3 axes
pression sur l'écran	ecran LCD tactile	5 boutons
secouement du cube	accéléromètre 3 axes	accéléromètre 3 axes
affichage	ecran LCD	PaPiRus ePaper

Pour manipuler la Raspberry pi Zero et ePaper PaPirus comme un objet tangible, un problème préliminaire a été soulevé : le Sifteo possède sa propre autonomie de batterie et il doit être proche d'un ordinateur pour pouvoir y jouer. Or, la Raspberry Pi Zero est un micro-ordinateur qui doit pouvoir être allumé soit via un ordinateur, soit via une prise ou une batterie. Cela signifie que pour manipuler plusieurs Raspberry Pi connectées avec une ePaper, nous devons trouver un moyen d'allumer la Raspberry sans câble extérieur.

Pour résoudre ce problème, nous devons ajouter le composant batterie externe afin de garder la Raspberry Pi allumée durant l'utilisation. Cela fait un composant de plus à ajouter à la Raspberry Pi. Pour la suite, nous prenons l'hypothèse que la Raspberry est allumée sans câble extérieur.

4.4.1 Fonctionnalité : contact entre cubes voisins

Le Sifteo cube possède son propre protocole NFC. Il s'agit d'une technologie qui permet d'échanger des données entre deux périphériques sur une courte portée avec une communication sans fil. C'est pourquoi plusieurs propositions de solutions ont été apportées. Passons en revue quelques solutions et analysons-les.

Le principal but est de simuler la connexion entre les deux cubes. Ils doivent pouvoir s'échanger des informations dès qu'ils sont en contact l'un avec l'autre. Le Sifteo cube possède des capteurs aux quatre côtés.

Plusieurs hypothèses peuvent être envisageables :

Module NFC avec carte

La première est d'utiliser un module NFC avec carte afin d'identifier les cubes qui sont mis en contact l'un avec l'autre. Lorsque les deux cubes seront identifiés, ils auront la possibilité de s'envoyer des données à travers le module NFC ou via le wifi. En effet, la Raspberry Pi Zero possède un module Wifi mais ce n'est pas toujours le cas. Le modèle de communication sera du pair-to-pair. Cela permet à chaque entité d'être à la fois client et serveur.

Pour utiliser cette solution, la Raspberry doit en même temps posséder ce module ainsi que la carte à puce, idéalement sur les quatre côtés du cube afin de savoir de quel côté le cube communique. Pour réaliser un cube, nous devons donc obtenir quatre modules, un par côté. Cette solution, peu maniable, pourrait poser un problème lors de l'expérience utilisateur.

Utilisation de câbles USB

Une deuxième solution pour envoyer les données serait l'utilisation de câbles comme des USB. Cette alternative comporte des problèmes puisqu'on devrait connecter les cubes manuellement. Le jeu en deviendrait moins attrayant. De plus,cette solution ne semble pas envisageable avec la Raspberry Pi Zero car elle nécessiterait une intervention externe avec un joueur lançant la commande d'envoi des données.

Boutons pressoirs

Une troisième hypothèse intéressante mais elle risque de poser problème pour d'autres fonctionnalités : l'utilisation des boutons-pressoirs pour simuler une connexion entre les deux cubes. Par exemple, pousser le bouton 1 d'un cube et le bouton 5 d'un autre cube en même temps permettrait de savoir que ces deux-ci souhaitent communiquer. Cependant, l'option des boutons est déjà prévue pour interagir avec l'écran.

Capteurs infrarouges

Une autre hypothèse serait d'utiliser un capteur infrarouge permettant de détecter la présence d'un objet. En effet, le capteur émet un faisceau lumineux en continu et lorsqu'un objet se trouve sur son chemin, la lumière est interrompue. Cette méthode permet de détecter un cube à une proximité d'un millimètre. Cependant, une fois détecté, nous devons trouver un moyen pour que les deux cubes se reconnaissent mutuellement afin de s'échanger des données. Un autre inconvénient de cette solution est que ce capteur peut détecter un objet à une distance jusqu'à 10 cm. Dans ce type de jeux, cela reviendrait à

détecter tous les cubes même de manière non intentionnelle.

Après avoir résolu le problème du contact entre les deux cubes, nous devons aussi permettre aux cubes de se reconnaître entre eux grâce éventuellement à des identifiants et des rôles.

4.4.2 Fonctionnalité : inclinaison et flip

Pour rappel, le Sifteo cube possède un accéléromètre permettant d'évaluer l'inclinaison du cube par rapport aux trois axes.

Pour gérer la réaction à l'inclinaison de notre "cube" (Fig 4.3), il est nécessaire que la Raspberry soit connectée en même temps à ePaper et au module MPU-9250 (l'accéléromètre). L'accéléromètre permettra de calculer l'inclinaison du module PaPiRus. Nous souhaitons reconnaître les quatre indicateurs (gauche, droite, avant, arrière) par rapport à la position neutre.

Pour cela, nous calculerons les différentes coordonnées (Fig 4.5) de l'accélération du module afin de déterminer nos quatre indicateurs. En étudiant les données,nous pourrions établir des intervalles de données identifiant une position du PaPiRus. Pour la réalisation du prototype, ce sera un élément important de traiter ces données.

Par ailleurs, sur le marché, un autre capteur trois axes pour Raspberry appelé inclinomètre permet aussi de mesurer précisément un angle ou l'inclinaison d'un objet. Cependant, l'accéléromètre propose plus de fonctionnalités que l'inclinomètre.

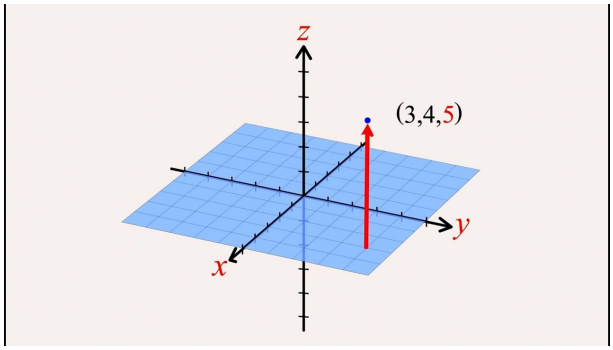

FIGURE 4.5 – Système d'axes de l'accéléromètre

4.4.3 Fonctionnalité : secouement

L'accéléromètre permet aussi de mesurer l'effet secouement. Nous pourrons utiliser les données récoltées afin d'établir comme pour l'inclinaison un indicateur nous permettant de découvrir si le cube a été secoué ou non . Malheureusement comme précisé plus haut, l'écran PaPiRus n'est pas adapté à des rafraîchissements trop rapides. L'idée de visualiser sur l'écran l'effet de secouement en temps réel n'est donc pas possible. Il faut donc pouvoir traiter les données capturées directement afin de permettre à l'écran de réagir lorsqu'il détecte un secouement. De plus, la Raspberry n'est pas faite pour réaliser des calculs en temps réel. Nous suggérons d'utiliser cette fonctionnalité dans le but de réinitialiser les données du cube par défaut.

4.4.4 Fonctionnalité : boutons pressoirs

Rappelons que le Sifteo possède un écran couleur LCD tactile permettant de voyager à travers le menu et d'avoir une certaine réaction de sa part. Malheureusement, l'écran ePaper n'est pas tactile et surtout très fragile. Il est donc impossible de reproduire cette fonctionnalité à la perfection. Cependant, ePaper dispose de cinq boutons sur le côté donc chaque pression permet d'activer une portion de code ce qui nous permet de simuler l'écran tactile. L'hypothèse de solution pour la gestion de la pression de l'écran est d'utiliser quatre de ces boutons afin d'obtenir des possibilités différentes. Le cinquième peut être utilisé comme "effet" retour ou "effet" erreur en fonction du jeu.

4.4.5 Gestion de l'affichage

Pour l'affichage, nous allons utiliser un écran ePaper PaPiRus qui simule l'effet encre de Chine. Le Sifteo cube est un écran couleur LCD tandis que ePaper PaPiRus est monochrome. Les images sont adaptées à la taille de l'écran. Les images doivent être les plus simples afin de bien distinguer les reliefs. Etant donné que l'écran consomme peu d'énergie, une image avec beaucoup de contrastes sera mal affichée par l'écran. Il faut faire attention à ce détail si l'on veut créer un jeu nécessitant beaucoup d'animations. Le ePaper n'est pas capable de gérer dynamiquement un flux d'images comme une mini vidéo. En revanche, il est capable de réaliser une petite animation avec plusieurs images qui se suivent comme un gif animé. PaPiRus est apte à dessiner des formes géométriques telles que le cercle, le rectangle.

L'écran ePaper a la caractéristique de ne pas être rafraîchi une fois le programme terminé et/ou l'ordinateur éteint. Cette option peut être utile lorsqu'on souhaite obtenir un résultat final (obtenir une phrase composée de plusieurs ePaper les uns à la suite des autres). Cependant, après un certains temps, l'écran s'efface.

L'écran ePaper n'est donc pas adapté à n'importe quel type de jeu. Par exemple, il n'est pas conçu pour un jeu tel que Fat and Furious [14]. En effet, il présente beaucoup de couleurs et d'animations. De plus, la gestion des différentes actions doivent se résoudre rapidement. Le changement d'affichage d'un ePaper prend un certain laps de temps.
Cependant, il est adapté à un jeu tel que mon prototype, expliqué plus loin dans ce chapitre. Ce jeu est avant tout basé sur la réflexion et non la vitesse d'exécution.

4.5 Prototype non-fonctionnel

4.5.1 Scénario de jeu

Le but de ce scénario global d'utilisation est de comprendre la notion de fonction informatique. Rappelons que les fonctions exigent la capacité d'anticiper une situation dans sa tête : un cheminement entre l'input de la fonction et son output. Nous avons élaboré un jeu de dominos grâce auquel les étudiants devraient acquérir les différents niveaux d'abstraction qui mènent aux fonctions.

Nous supposerons que les fonctionnalités de communication, d'échange de données et d'inclinaison marchent correctement. Voici comment pourrait se dérouler le jeu : d'abord, les étudiants recevront l'énoncé suivant :

" Namur compte parmi ses entreprises, l'entreprise "creeTonTshirt", l'entreprise "personnaliseTonTshit", l'entreprise "livreTout", l'entreprise "publieTout" et l'entreprise "venteTshirt".
L'entreprise "creeTonTshirt" est une entreprise textile qui produit des T-shirts. La commande peut être effectuée par un particulier ou un grossiste. Si le client donne uniquement la quantité de t-shirts qu'il souhaite, l'entreprise crée ce nombre de T-shirts. Le t-shirt par défaut est en polyester, manches courtes, blanc et col rond. Si le client souhaite un textile particulier exemple "coton", il doit le spécifier en plus de la quantité, de même si le client souhaite un certain type de manches ou un certain type de cols. Si le client souhaite personnaliser son T-shirt par un logo ou un texte, l'entreprise "creeTonTshirt" doit faire appel au service de son sous-traitant : l'entreprise "personnaliseTonTshit".
L'entreprise "personnaliseTonTshit" est une entreprise "B to B" ou "B to C". Le client doit fournir le ou les T-Shirts à personnaliser d'un logo ou d'un texte.
L'entreprise " publieTout" est une entreprise de publicités à la disposition des entreprises. Le client envoie un texte et l'entreprise "publieTout" publie ce texte sur tous les réseaux sociaux.
L'entreprise "livreTout" possède une flotte de camions. Une entreprise cliente peut les appeler pour livrer sa production à un grossiste. Un camion de livreTout vient chercher la production et livre celle-ci au client spécifié.
L'entreprise "venteTshirt" met à l'étalage les différents T-shirts qui lui sont livrés."

Chaque entreprise citée ci-dessus est représentée sur un cube, idem pour les inputs (quantité de T-shirt, logo, type de textile, texte etc) et les outputs (T-shirts personnalisés etc). le but pour l'étudiant est d'imaginer des enchaînements qui fonctionnent. La Fig 4.7 correspond à une solution incorrecte.

Trois parcours pédagogiques de niveaux différents sont proposés ci-dessous :

Premier parcours :

Règle du jeu : le programme énonce une situation d'appel à une entreprise et l'étudiant est amené à la représenter sous un format dominos. Ensuite, il devra écrire les spécifications correspondantes. Par exemple :

Je commande à l'entreprise "creeTonShirt" 10 T-shirts avec des manches longues. Voir (Fig 4.6)

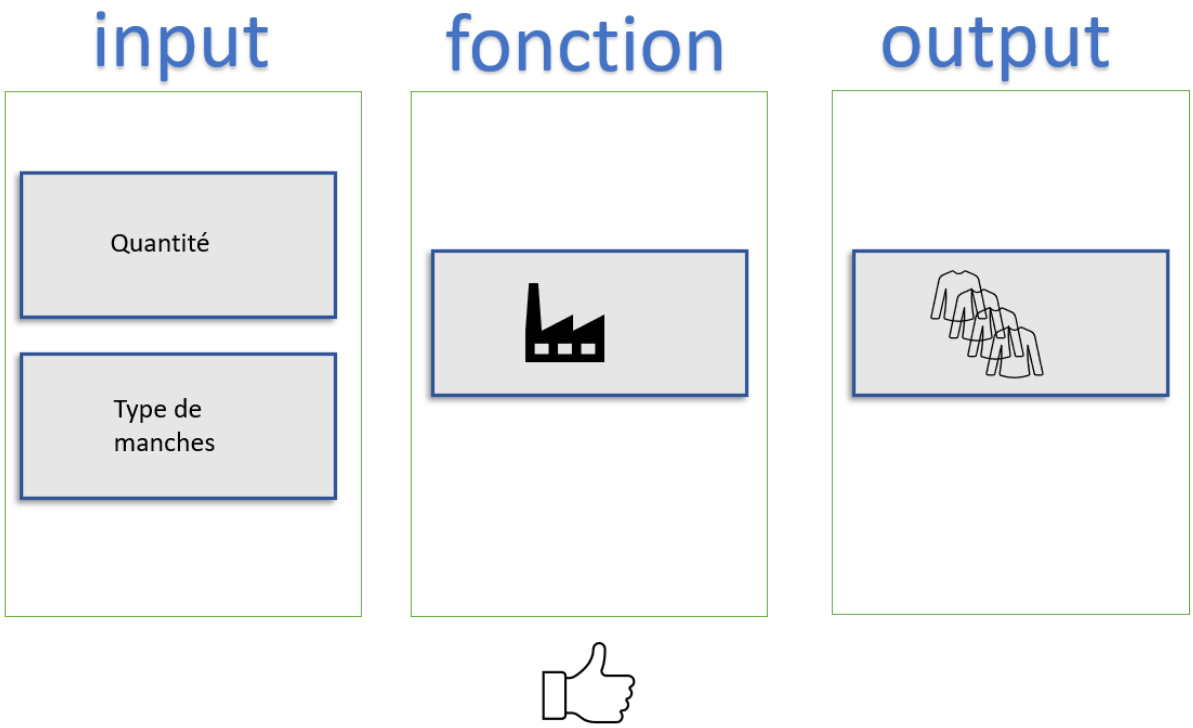

FIGURE 4.6 – Parcours 1 - un exemple correct

Voici deux exemples de spécifications selon le niveau.

Le première écrite en français :

Fonction de l'entreprise "creeTonTshirt" : crée des T-shirts selon les demandes de l'utilisateur. Il peut définir une quantité, spécifier le type de manches, le type de col ainsi que la couleur. S'il donne juste une quantité, il aura le T-shirt basique c'est-à-dire le T-shirt manche courte blanc, col rond dans la quantité demandée.
Inputs : la quantité (entier positif) - type de manches
Output : liste de T-shirts.

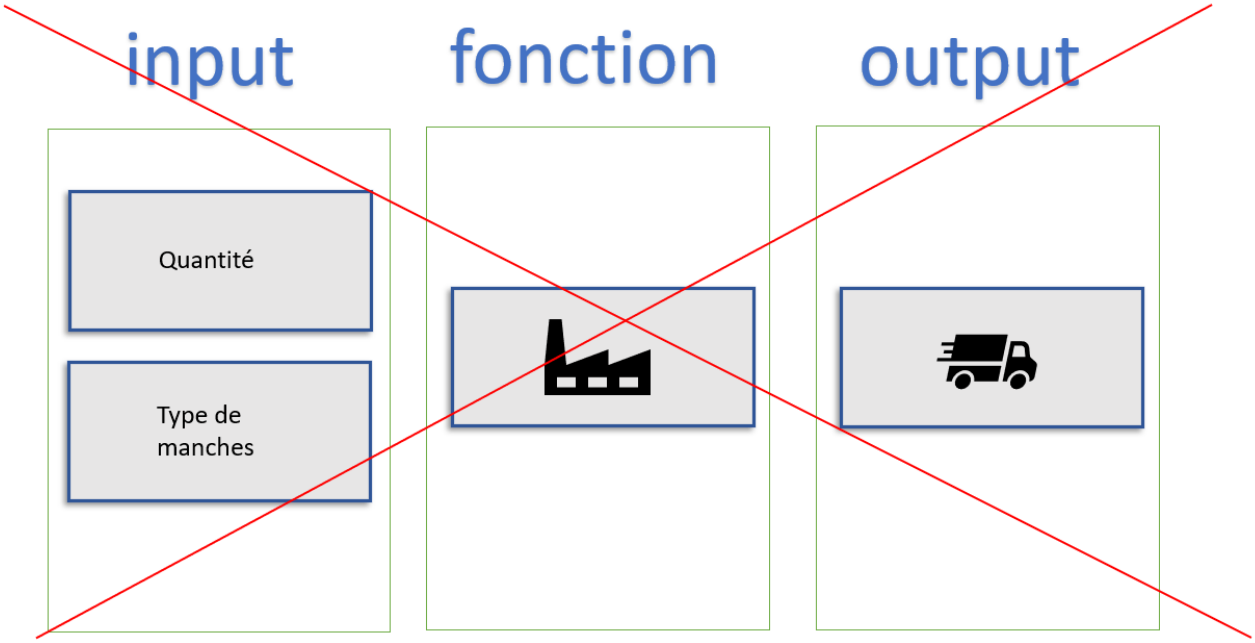

FIGURE 4.7 – Parcours 1 - un exemple incorrect

Le deuxième écrit dans un langage :

```
def creeTshirt(nombre, mancheLongue):
    '''
    fonction permettant de produire un nombre de T-shirts manches longues.
    intput: nombre(Int) - mancheLongue(String)
    output: liste de t-shirts
    '''
    for(int i = 0; i < nombre; i++)
        create = "mancheLongue"
        shirts.add(create)
    return shirts


""" Appel """
shirt = createTshirt(1, mancheLongue)
```

L'étudiant sera félicité si la suite est correcte. D'un point de vue pédagogique, en référence à la pyramide de Bloom, ce jeu n'empêche pas l'essai-erreur ce qui permet aussi à l'étudiant de faire son apprentissage. Ce type de parcours permet de travailler les niveaux compréhension et application de la taxonomie de Bloom.

Deuxième parcours :
Règle du jeu : créer une suite de dominos qui représente une situation concrète possible c'est-à-dire une situation qui évoque l'appel à une entreprise en lui donnant les bons paramètres d'entrée et en symbolisant correctement le ou les bons paramètres de sortie. Cette fois, la situation n'est plus énoncée mais c'est à l'étudiant de créer un maximum de situations correctes en un laps de temps déterminé. Lorsque le système détectera une situation correcte, il le fera savoir au joueur. "Bravo" apparaîtra sur un des cubes et le joueur aura un point. L'étudiant sera amené à photographier la situation. Lorsque le jeu sera terminé, il devra reprendre chaque photo, écrire l'énoncé de chaque situation puis écrire les spécifications de la fonction correspondante (de manière plus ou moins formalisée en fonction des niveaux).

Par exemple :

Je commande un T-shirt par défaut à l'entreprise "creeTonShirt". J'aimerais ensuite personnaliser en rajoutant un logo Voir (Fig 4.8)

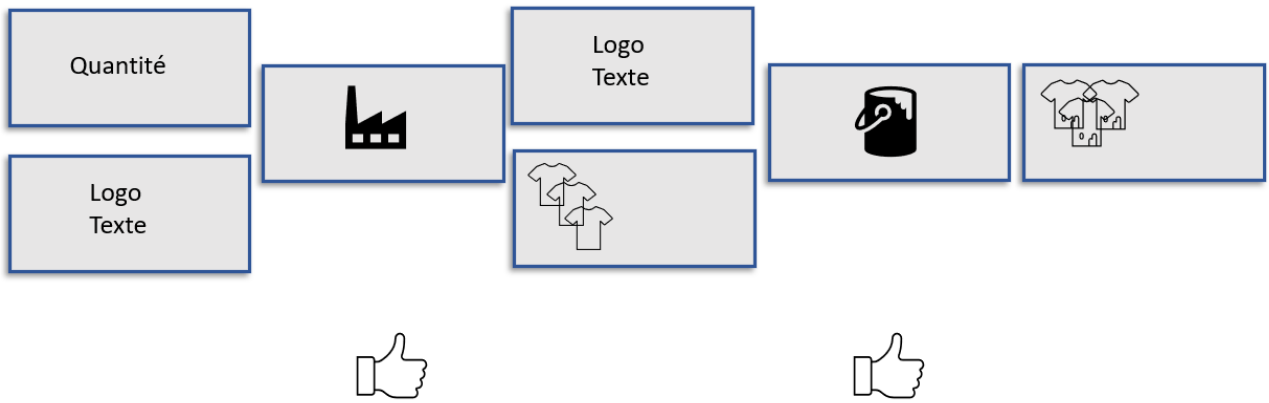

FIGURE 4.8 – Parcours 2 - un exemple correct

```
def createTshirt():
    '''
    un T-shirt par standard :
    blanc, manche_courte, polyester , col_v
    '''


def createTshirt(nombre):
    '''
    fonction permettant de produire un nombre de Tshirt
    standard: blanc, manche_courte, polyester , col_v
    intput : nombre(int)
    output : nombre de T-shirt
    '''
    for(int i = 0; i < nombre; i++)
```

```
        create = blanc + manche_courte + col_v + polyster
        shirts.add(create)
    return shirts



def createTshirt(nombre, mancheLongue):
    '''
    fonction permettant de produire un Tshirt
    intput: nombre(int) - nombre de fois que l'on souhaite le Tshirt
    intput : mancheLongue - type de manche
    output: liste de T-shirts manches longues
    '''
    for(int i = 0; i < nombre; i++):
        create = blanc + mancheLongue + col_v + polyster
        shirt.add(create)
    return shirts



def personnaliseTonTshirt(shirts, logo, texte):
    '''
     fonction permettant de personnaliser un Tshirt
     intput : logo - l'image sur le T-shirt
     intput : texte - le texte sur le T-shirt
     output : shirts: liste de T-shirts avec le logo et le texte
    '''
    return "shirts"+"logo"+"texte";



""" Appel """
shirt = createTshirt()
personnaliseTonTshirt(shirt, logo, texte)
```

Ce type de parcours permet de travailler le niveau "application" et "analyse" de la taxonomie de Bloom. L'étudiant s'initiera au formalisme des spécifications des fonctions. Il peut faire des essais-erreurs mais s'il veut aller vite, il doit commencer par anticiper la solution et donc analyser le problème .

Troisième parcours :
Régle du jeu : Le jeu se déroule de la même façon que le deuxième parcours. Cependant, l'étudiant devra énoncer la situation avant de la réaliser. Cette nouvelle règle ne permet plus l'essai-erreur et oblige donc l'étudiant à anticiper un déroulement dans sa tête. Ce type de parcours a pour but de travailler les niveaux "analyse" et "synthèse" de la pyramide de Bloom. Pour travailler la compétence supérieure de la taxonomie, l'évaluation, chaque joueur de la même équipe devra valider la situation avant de la concrétiser.

4.5.2 Technique

La gestion des différentes fonctionnalités dépendra du type de jeu choisi. Les tableaux Fig3.1 et Fig3.4 présentés précédemment permettent d'avoir une idée des différents actions de chaque fonctionnalité.

Nous choisirons par exemple le secouement pour réinitialiser le cube par défaut. Les boutons-pressoirs permettront de changer de texte ou d'images. L'inclinaison permettra aux cubes de définir s'ils sont des cubes "input" ou de "output". Le contact entre voisins permettra de valider un "input" ou un "output". Pour gérer le contact et le transfert de données, la Raspberry devra au moins avoir des caractéristiques telles qu'un identifiant afin de la reconnaître.

Dans le chapitre suivant, nous présenterons les cas d'utilisation ainsi que le scénario de chaque fonctionnalité et nous expliquerons les différentes facettes de l'implémentation.

Chapitre 5

Réalisation

5.1 Introduction

Dans ce chapitre, tout d'abord, nous présenterons les différents cas d'utilisation. Ensuite, pour chaque fonctionnalité, nous établirons un scénario et enfin, nous expliquerons comment nous l'avons testée et implémentée.

5.2 Cas d'utilisation

Le diagramme des cas d'utilisation (Fig 5.1) permet de visualiser les différentes actions que l'acteur peut réaliser avec le programme.

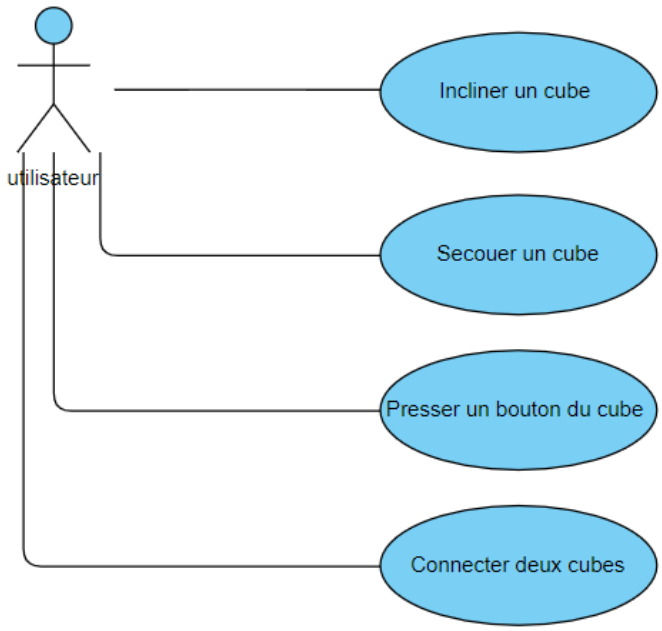

FIGURE 5.1 – Schéma des cas d'utilisation

5.3 Développement des cas d'utilisation

5.3.1 Fonctionnalité : contact entre cubes voisins

Cas d'utilisation et scénario par rapport au prototype

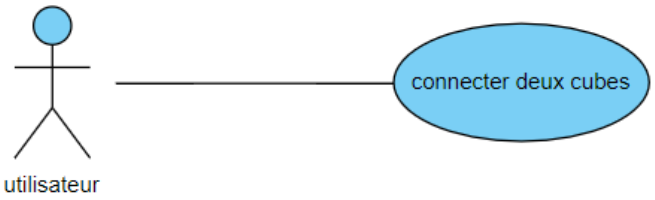

FIGURE 5.2 – Cas d'utilisation : connecter deux cubes

La connexion entre deux cubes	
Utilisateur	Système
ce use case commence lorsque l'utilisateur connecte un cube entrée avec un cube entreprise	le système va identifier les deux cubes connectés le système va déterminer si le cube entrée est bien à droite du cube entreprise ou le cube sortie à gauche
	si c'est le cas, il vérifie si les valeurs d'entrées sont utilisées par l'entreprise si ce n'est pas le cas, il affiche une erreur.

Implémentation

Nous n'avons pas pu réaliser et tester cette fonctionnalité par manque de matériel. En effet, nous ne disposions que d'une alimentation pour deux Raspberry. Pour tester le contact entre deux cubes (Raspberry), il nous fallait deux capteurs permettant de les identifier et de gérer leur proximité.

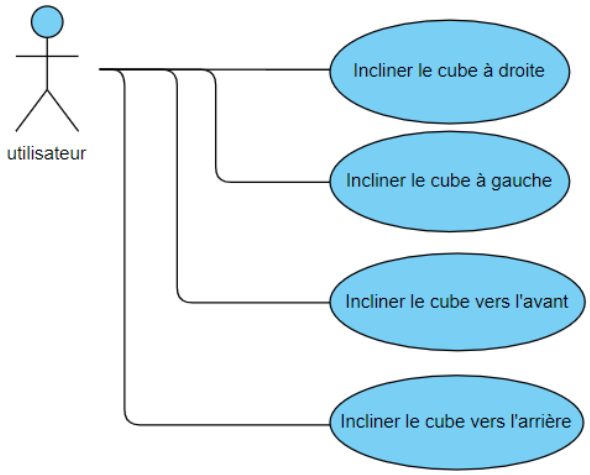

FIGURE 5.3 – Cas d'utilisation : incliner les cubes

5.3.2 Fonctionnalité : inclinaison et flip

Cas d'utilisation (Fig 5.3) et scénario par rapport à l'implémentation

L'inclinaison du cube	
Utilisateur	Système
ce use case commence lorsque l'utilisateur incline le cube vers la gauche	le système lui répond en affichant l'image Fig 5.5
l'utilisateur incline le cube vers la droite	le système lui répond en affichant l'image Fig 5.7
l'utilisateur incline le cube vers l'avant	le système lui répond en affichant l'image Fig 5.8
l'utilisateur incline le cube vers l'arrière	le système lui répond en affichant l'image Fig 5.4

Implémentation

Comme nous l'avons mentionné ci-dessus, c'est l'accéléromètre qui va nous permettre de gérer l'inclinaison du cube. Pour détecter le type de mouvement du cube ainsi que sa direction (gauche, droite, haut, bas), nous avons réalisé un script python nous permettant de récupérer les données de l'accéléromètre dans un fichier csv. Les données qui nous intéressaient étaient l'accélération en X , en Y et en Z. Pour ce test, l'accéléromètre a été placé de telle sorte que les axes soient orientés de la même façon que la Fig 4.5. En

analysant ces données, voir en annexe, nous avons constaté les points suivants :

- Angle 90° vers la droite : l'axe Y est négatif (passe du point neutre jusque -0.9 g) ;
- Angle 90° vers la gauche : l'axe Y est positif (passe du point neutre vers 0.9 g) ;
- Angle 90° vers l'avant : l'axe X est positif (passe du point neutre jusqu'à 0.9g) et l'axe Z est négatif ;
- Angle 90° vers l'arrière : l'axe X est négatif (passe du point neutre jusqu'à -0.9g) et l'axe Z est positif.

Ceci correspondait assez bien à nos attentes. Nous avons donc implémenté les conditions qui identifient les positions d'inclinaison.
Il nous est apparu lors de ces tests que le point neutre variait un peu mais que cela n'influençait pas nos résultats.
Lors des nos tests, l'une des images Fig 5.4 à Fig 5.8 devait s'afficher. Cela permettait de vérifier :

- que le capteur était bien connecté à la Raspberry, elle-même bien connectée au PaPirus ;
- que la lecture et l'affichage des images soient corrects ;
- que le type d'inclinaison était détecté correctement. Par exemple, si le capteur est penché vers la gauche, l'image Fig 5.5 est affichée.

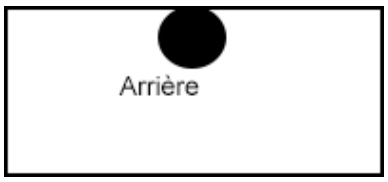

FIGURE 5.4 – Position arrière

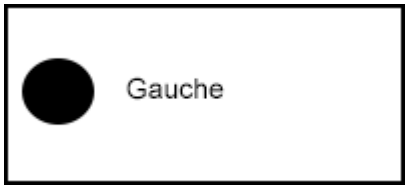

FIGURE 5.5 – Position gauche

FIGURE 5.6 – Position neutre

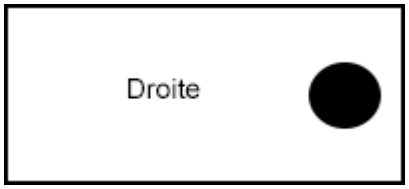

FIGURE 5.7 – Position droite

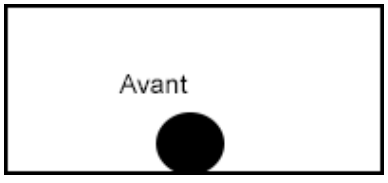

FIGURE 5.8 – Position avant

5.3.3 Fonctionnalité : secouement

Cas d'utilisation (Fig 5.9) et scénario par rapport au prototype

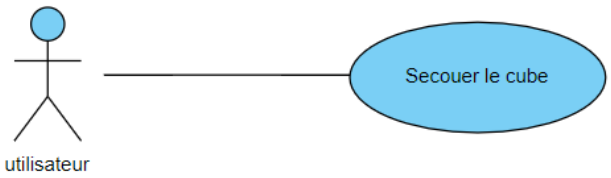

FIGURE 5.9 – Cas d'utilisation : secouement du cube

Le secouement du cube	
Utilisateur	Système
ce use case commence lorsque l'utilisateur secoue le cube	le système lui répond réinitialisant les informations par défaut

Implémentation

Nous avons utilisé la même démarche concernant la gestion du secouement. La récupération de plusieurs secondes de données a été réalisée afin de les analyser. Le capteur a été secoué de gauche à droite au niveau de la table. Cependant, des problèmes de câblage ont perturbé les essais mais nous avons quand même tenté d'exploiter certaines données. En effet, nous avons constaté que, si pendant plusieurs secondes, les valeurs en X et Y étaient similaires mais changeaient de signe d'une seconde à l'autre et que Z était proche de -1, nous devions détecter un secouement horizontal au niveau de la table.

Une autre façon de faire serait de détecter un mouvement de secouement par le fait que le cube passe régulièrement d'un coté à l'autre du point neutre. Les valeurs du point neutre varient à chaque test. Il serait donc nécessaire de calibrer correctement ce point.

Une dernière idée pour détecter le secouement serait d'utiliser la grande variabilité des données dans ce cas et de calculer la distance entre le triplet (ax,ay,az) du moment t et le même triplet du moment d'avant : $\sqrt{(x1-x0)^2+(y1-y0)^2+(z1-z0)^2}$.

5.3.4 Fonctionnalité : boutons pressoirs

Cas d'utilisation (Fig 5.10) et scénario par rapport au prototype

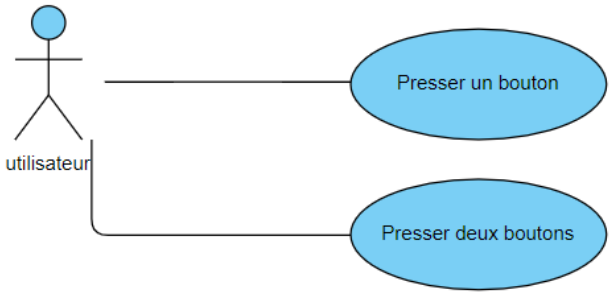

FIGURE 5.10 – Cas d'utilisation : pression sur des boutons

La pression des boutons	
Utilisateur	Système
ce use case commence lorsque l'utilisateur presse sur un bouton	le système lui répond en affichant le texte du bouton
l'utilisateur presse sur deux boutons	le système lui répond en affichant le texte présent sur les deux boutons

Implémentation

Etant donné que ePaper ne possède pas d'écran tactile, nous utilisons les cinq boutons intégrés à PaPiRus pour simuler la pression sur l'écran. Nous souhaitons pouvoir afficher différentes entrées et sorties. Lors de la réalisation, les cinq boutons ont été testés afin de déterminer la façon de les utiliser.

Nous avons implémenté la possibilité de combinaisons de deux boutons pressés en même temps. Pour le prototype ci-dessus, le cube possède cinq boutons avec cinq propositions d'arguments . Nous utilisons la combinaison de deux boutons pour simuler un input. Par exemple, le bouton cinq affiche "nombre" et le bouton quatre affiche "typeCol", si nous voulons utiliser les deux , nous exécuterons les boutons quatre et cinq en même temps, l'écran affichera donc "nombre + typeCol". Cette méthode permet durant le jeu de simuler deux entrées avec un cube. (Exemple Fig 5.14)

Nous nous sommes contentés de gérer juste deux arguments. C'est assez complexe de presser simultanément plus de deux boutons.

Les Fig 5.13, Fig 5.14, Fig 5.15 illustrent un exemple de changement d'entrées pour un même bouton. En effet, avant de presser sur les deux boutons (Fig 5.14), l'entrée du bouton trois affichait Fig 5.13 et juste après avoir pressé deux boutons simultanément le résultat est différent (Fig 5.15).

5.3.5 Gestion de l'affichage

Lors de la réalisation, nous avons implémenté l'affichage. Tout d'abord, le programme "papirus-animation" présent déjà sur PaPirus permet de gérer l'affichage d'un gif animé. Il faut remarquer que des images à contrastes élevés ne s'affichent pas correctement. Nous ne distinguons pas les différents éléments.

Ensuite, nous avons implémenté un affichage différent par bouton. Nous avons pu ainsi visualiser ce qu'il se passe lorsqu'on affiche plusieurs textes les uns à la suite des autres. Nous avons constaté que l'écran n'effaçait pas complètement les anciens textes. Nous pouvons le voir sur les figures (Fig 5.11 et Fig 5.12), le fond est légèrement grisé. Pour pallier ce problème, chaque fois que nous voulons afficher un texte, nous effaçons le ePaper avec la commande "clear". Le programme prend quelques secondes au moment où il exécute la commande "clear" et la commande "write". Relevons aussi que le ePaper ne supporte pas les accents, ni autres caractères spéciaux puisqu'il est programmé en anglais. Ils ne s'affichent tout simplement pas ou provoquent une erreur sur la console. En ce qui concerne les phrases, ePaper affiche simplement ce qu'il est en mesure d'afficher et ne fait pas défiler le texte. Il tronque tout simplement les phrases.

Dans le chapitre suivant, nous faisons le point de cette recherche en présentant les difficultés rencontrées. Nous expliquerons ensuite l'avancement des différentes fonctionnalités faites, réussies, manquantes et à améliorer. Nous proposerons des suggestions utiles pour les futurs travaux.

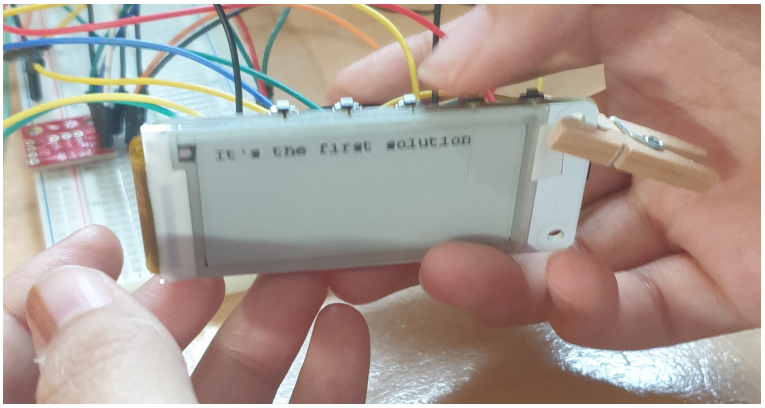

FIGURE 5.11 – Affichage de texte lors d'une pression bouton

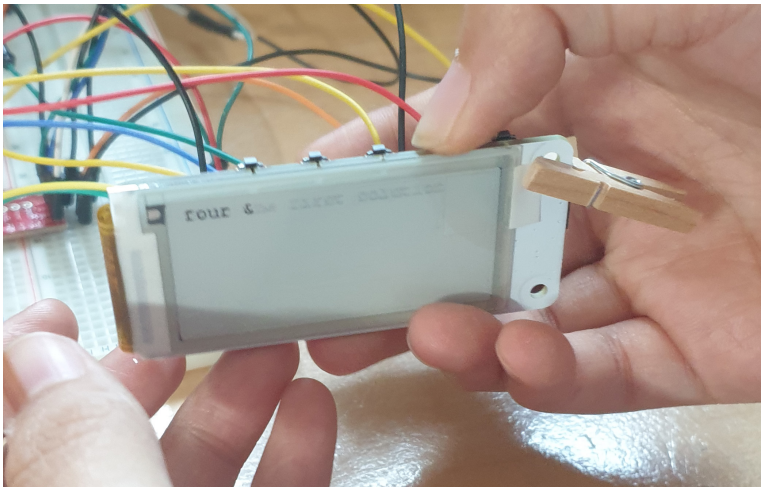

FIGURE 5.12 – Affichage de texte lors d'une pression bouton

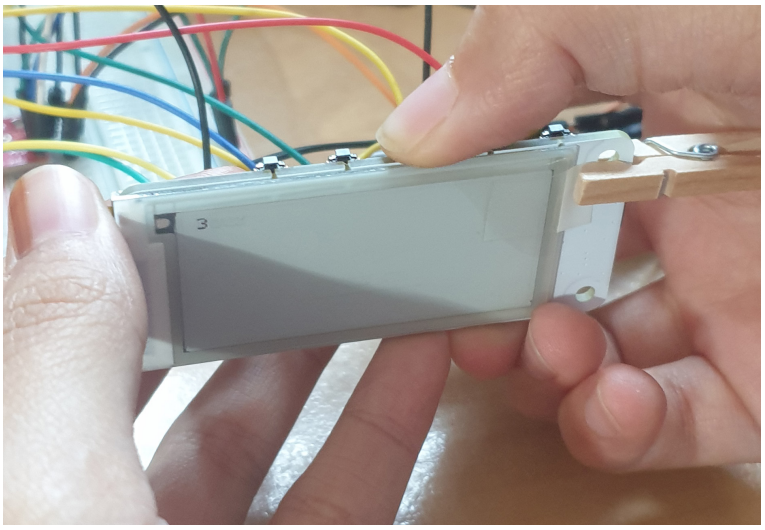

FIGURE 5.13 – Affichage de texte lors d'une pression bouton

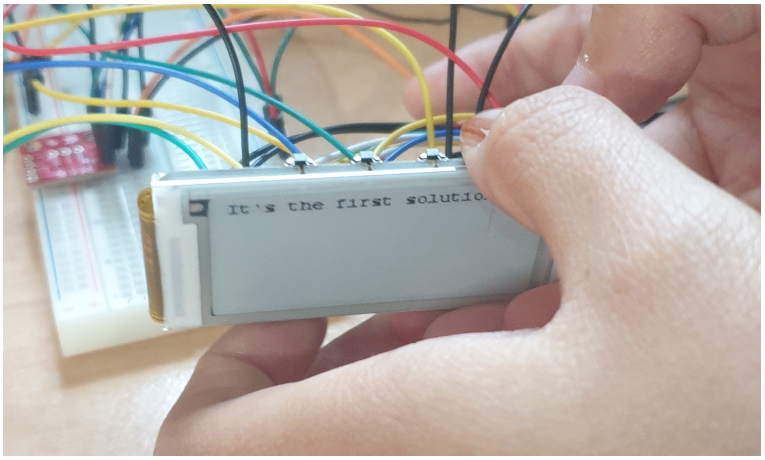

FIGURE 5.14 – Affichage de texte lors d'une pression sur deux boutons

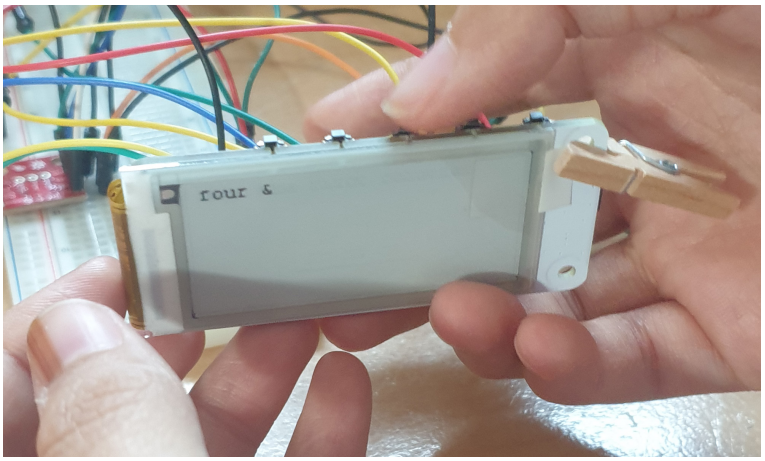

FIGURE 5.15 – Affichage de texte lors d'une pression bouton

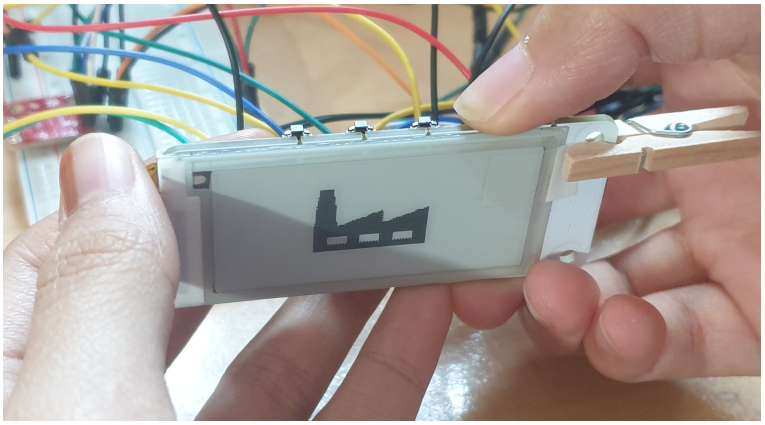

FIGURE 5.16 – Affichage image lors d'une pression bouton

Chapitre 6

Discussion

Ce chapitre présente les différentes difficultés rencontrées tout au long de la réalisation de ce travail ainsi que d'éventuelles suggestions d'amélioration pour les travaux futurs.

6.1 Difficultés rencontrées

Dans cette section, nous parlerons des plusieurs fonctionnalités que nous avons pu tester avec les différents outils disponibles : à savoir, l'affichage, la réaction à la pression des boutons et la réaction à l'inclinaison. Les autres fonctionnalités telles que le contact entre cubes voisins, la réaction face à un secouement de l'appareil n'ont pas pu être testées ou du moins pas complètement.

6.1.1 Composants électroniques

Pour démarrer le projet, il fallait pouvoir faire interagir l'accéléromètre et l'ePaper avec la Raspberry Pi Zero. Il a fallu câbler chacun des composants, voir annexe (Fig 8.2) avec la Raspberry. Durant la réalisation, nous avons remarqué que certains pins étaient utilisés par les deux composants. Le ePaper utilise tout les pins de la Raspberry, il n'est pas possible d'utiliser d'autres composants simultanément. Nous nous somme servis d'un "breedboard" afin de pouvoir connecter d'autres capteurs sur la Raspberry. Nous avons connecté les deux composants à la Raspberry en utilisant des "jumpers mâles-femelles". Cette technique présentait un inconvénient majeur : à tout moment, les pins pouvaient se déconnecter.

De plus, pour tester l'accéléromètre, il fallait que certains pins soient mis en contact avec le composant en continu. Cela a nécessité beaucoup de câbles juste pour ajouter deux composants. Ainsi, pour réaliser les tests avec le capteur accéléromètre, nous avons souvent rencontré des problèmes de câblage et d'interconnexion.

6.1.2 Fonctionnalités

Venons-en maintenant aux problèmes liés aux fonctionnalités. Rappelons que les fonctionnalités testées et implémentées sont les suivantes : l'affichage, la pression des boutons, la réaction à l'inclinaison et le secouement.

Pour l'affichage, nous n'avons pas eu de grandes difficultés techniques. Lors des tests, nous avons simplement déduit quelques constatations :

— pour les images, il est recommandé d'utiliser celles présentant peu de contrastes et de reliefs ;
— une animation d'un gif en continu peut faire bloquer le déroulement du jeu ;
— pour l'affichage d'un texte, c'est la police d'écriture qui détermine le nombre maximum de caractères à afficher sur l'écran.

D'un point de vue technique, la gestion des boutons était en soi facile. Cependant, pour notre jeu, nous souhaitions que les boutons interagissent différemment en fonction de l'évolution du jeu. Nous voulions aussi pouvoir utiliser plus de cinq inputs pour un même "cube" en utilisant différentes combinaisons de boutons. Il a fallu, dès lors, les implémenter sachant que chacune d'elles devrait correspondre à un code différent à chaque moment du jeu.

Pour travailler avec l'inclinaison et le secouement du cube, nous devons utiliser les données générées par le script Python provenant du capteur. Il ne nous a pas été possible de définir la position neutre de manière précise. Les données changeaient fortement d'un test à un autre. Nous avons quand même pu définir les inclinaisons (par rapport à la table) excepté le secouement mais de toutes façons, il faudrait pouvoir redéfinir toutes ces positions de manière plus constante.

6.2 Réalisations, améliorations et suggestions

Dans cette section, nous présenterons ce qui a fonctionné, nous définirons des pistes d'amélioration et nous attirerons l'attention sur quelques points essentiels au bon déroulement de futurs travaux.

6.2.1 Composants électroniques

Tout d'abord, les composants devraient être soudés afin d'empêcher toutes les interruptions de connexion et de rendre les données plus constantes d'un test à l'autre. Ci-dessous, nous présentons une série de composants électroniques pouvant être utile à la réalisation d'un cube tangible . En plus de cette liste, nous avons besoin d'autres matériels tels que les différents câbles "mâles-mâles", "femelles-femelles", "mâles-femelles", un GPIO hammer, un "breedboard" etc... permettant de les connecter correctement.

Voici une idée de composants utiles en fonction des différentes fonctionnalités visées.

Composants utiles pour les fonctionnalités	
Fonctionnalités	Composants
contact entre voisins	capteurs NFC/ module Wifi/ capteurs infrarouges
affichage	écran LCD tactile / ePaper PaPiRus
effet inclinaison / flip	accéléromètre 3 axes / inclinomètre
effet secouement	accéléromètre 3 axes
pression écran	ePaper PaPirus et ses boutons
	batterie (afin de ne pas avoir de Raspberry câblée)
Autres suggestions	
feedback auditif	mini-haut parleur
autre feedback visuel	module LEDS

Méthodologie pour les projets futurs

Nous rappelons ici la démarche d'analyse nécessaire pour déterminer la composition de tout nouveau dispositif qui se veut être le transfert d'un ancien (comme le Sifteo par exemple). Tout d'abord, il faut prendre le temps de bien analyser l'utilité de l'ancienne formule et en déduire précisément les fonctionnalités intéressantes. Ensuite, nous devons comprendre les différentes technologies utilisées par l'ancien. Après cela, il est nécessaire d'étudier la documentation pour les comparer à d'autres technologies susceptibles d'atteindre le même objectif. A ce moment seulement, un choix peut être fait et les tests peuvent commencer. Si les tests sont concluants, le dispositif peut enfin être assemblé.

6.2.2 Fonctionnalités et jeu

Passons en revue les différentes fonctionnalités du jeu :

Pour les fonctionnalités d'inclinaison et de secouement, même si nous avons réussi à détecter et tester un pivotement de 90 degrés en prenant la table comme point de repère, il est clair qu'il nous faut un calibrage plus précis. La gestion du câblage proposée ci-dessus permettrait d'améliorer grandement ces fonctionnalités. En effet, grâce à des données plus précises, nous pourrions aisément calibrer un point neutre et mesurer l'évolution des données, ce qui nous permettrait d'implémenter plus rigoureusement les inclinaisons et les secouements.

En ce qui concerne l'affichage et les boutons pressoirs, nous avons réussi à implémenter l'affichage d'images ou de textes en fonction de différentes combinaisons de boutons. Il faut cependant être attentif à la simultanéité lorsque plusieurs boutons doivent être pressés. Le prototype non-fonctionnel proposé ci-dessus ne nécessite pas d'images avec de grands contrastes, ni de couleurs.

Par ailleurs, le contact entre cubes voisins est un élément clé qui nous manque. Plusieurs hypothèses ont été proposées pour détecter le cube voisin. N'ayant qu'un seul dispositif, aucun test n'a pu être réalisé à ce sujet.

Pour la réalisation d'un jeu complet, nous devons garder en mémoire que chaque cube est l'équivalent d'un nano-ordinateur. Les cubes n'exécuteront pas les fonctionnalités de la même manière. Si nous prenons l'exemple du prototype, nous définirions trois rôles clés : input, output et entreprise. Chaque Raspberry aura un rôle parmi les trois. Il est donc important de créer une structure permettant de donner un rôle précis à chaque cube, ainsi cela débloquera les fonctionnalités. Ce problème devra être analysé en fonction du jeu proposé.

6.3 Covid-19 : déplacements

Comme indiqué dans la section précédente, une étape importante d'analyse a dû être réalisée avant de commencer les tests. L'analyse a d'abord porté sur les fonctionnalités ensuite sur les hypothèses technologiques adaptées à celles-ci. Des achats en ligne nous ont permis de nous procurer un accéléromètre et un kit de soudure (GPIO hammer solderless, pour PaPiRus) au bout de trois semaines. N'ayant jamais eu de formation sur du matériel autre que software durant mon cursus scolaire, pour le choix du matériel électronique (carte SD - capteurs NFC - batterie - GPIO pins etc), il aurait fallu avoir la possibilité de rencontrer un spécialiste en magasin. Ceci n'a pas été possible car à ce moment les problèmes de déplacement (confinement) ont commencé.

Afin d'avoir une idée de l'intérêt de ce dispositif et du scénario global d'utilisation, l'idéal aurait été de réaliser une expérience auprès des étudiants ciblés : en l'occurrence, les étudiants de première année de baccalauréat. Le prototype n'étant pas fonctionnel, l'idée serait de réaliser l'expérience avec des petits cartons symbolisant un cube. Après avoir expliqué toutes les règles de fonctionnement du prototype, les étudiants jouent et manipulent les objets. Ensuite, nous recueillerions les critiques des utilisateurs pour avoir une idée de l'impact du produit et ainsi améliorer les fonctionnalités et/ou le prototype. Cette démarche nous permettrait de répondre aux questions suivantes :

- Ont-ils compris le but de l'exercice ?
- Ont-ils compris les concepts que nous voulions leur apprendre ?
- Est-ce difficile ? Trop facile ?
- Quelles sont les différentes remarques apportées ?
- Seront-ils capables de mettre en pratique l'idée véhiculée ?
- Comment appréhendent-ils la technologie ?
- Comment peut-on améliorer la technologie ?

Chapitre 7

Travaux futurs

Dans le chapitre précédent, nous avons discuté des difficultés rencontrées lors de la réalisation du prototype et nous avons apporté diverses suggestions d'amélioration. Dans celui-ci, nous proposerons d'autres pistes utiles pour compléter le projet. Rappelons que nous souhaitons utiliser les interfaces tangibles afin d'aider les étudiants à comprendre les notions d'abstraction.

Etudes et tests utilisateur

Tout d'abord, nous devons mesurer l'impact de cette solution. Comme dit précédemment, un test utilisateur devra cibler les étudiants de première année. Grâce à cela, nous pourrons répondre à diverses questions et adapter ainsi le projet. Cette enquête améliorera le prototype et les fonctionnalités. Cependant, il ne permettra pas de déterminer l'impact positif sur les capacités d'appréhension de l'étudiant.

L'idée serait de réaliser d'autres études, comme par exemple :
— échantillonner un groupe d'étudiants utilisant les applications tangibles telles que notre prototype et suivre leur évolution d'apprentissage durant au moins deux ans ;
— comparer avec un groupe témoin qui ne bénéficie pas de cette méthode d'apprentissage. Une étude [19] avait déjà été réalisée, malheureusement il ne mesurait pas l'impact dans le temps.

Les deux prochains points exposent des travaux techniques permettant de compléter correctement le prototype commencé.

Implémentation fonctionnalité "contact" et intégration de plusieurs cubes

Ensuite , dans ce travail, nous avons étudié la possibilité de réaliser un cube dans le but d'apprendre et de comprendre les notions de programmation. Dans un des projets futurs, nous devrions nous focaliser sur l'intégration de plusieurs nano-ordinateurs connectés les uns aux autres et déterminer si, ensemble, ils peuvent réaliser un jeu adéquat. Pour ce faire, nous devrions implémenter et intégrer la fonctionnalité "contact entre voisins".

Traitement des données des capteurs

De plus, lors de la réalisation, nous avons eu un problème de calibrage des données lorsque nous voulions tester les fonctionnalités d'inclinaison et de secouement. D'un point de vue technique, il serait intéressant de récolter des données à plusieurs endroits. Cela permettrait ainsi de les traiter et en retirer des résultats tels que des constantes. Cela permettra de déterminer avec précision les mouvements en fonction de leur position dans l'espace. Lors de ce projet, un travail préliminaire a été commencé sur le capteur accéléromètre. Il faudrait aussi étudier les autres données.

Ecran d'affichage

Notre premier choix s'est porté sur le ePaper PaPiRus, par son prix, son design, sa basse consommation et sa facilité d'intégration. Il est important de souligner qu'un tel écran possède aussi des inconvénients. En effet, comme précisé précédemment, cet écran ne supporte que les couleurs monochromes et ses capacités de rafraîchissements s'exécutent lentement. Le choix de l'affichage dépendra du style de jeu choisi.

Chapitre 8

Conclusion

Dans un premier temps, le début du travail a consisté en une recherche préliminaire au sujet des Sifteo dans le but de dégager une question précise. Suite à cela, l'objectif du projet est devenu : "Est-il possible de créer un Sifteo 2.0 dans le but d'aider les jeunes à apprendre et comprendre les notions de programmation, plus particulièrement les fonctions ? ".

Avant de se lancer dans l'analyse du problème, une étude des différents travaux concernant les Sifteo cubes a permis de comprendre le but de leurs applications et de lister leurs fonctionnalités : détecter un cube voisin, secouer un cube, faire pivoter un cube, presser l'écran. Nous avons ensuite classifié les études par domaine d'activité. Ces articles nous ont aussi appris que cette technologie était très souvent utilisée à des fins d'apprentissage. En effet, nous avons constaté que les jeux permettaient de se faire des représentations mentales et d'acquérir des notions abstraites intéressantes. Le côté ludique du dispositif motivait beaucoup les utilisateurs.

Nous avons par ailleurs constaté que cette technologie n'était plus vendue sur le marché. Cela rendait donc impossible de fournir un scénario d'utilisation de cette technologie afin d'apprendre et comprendre les notions de programmation.

Cela nous a amené à nous questionner sur le point suivant : peut-on recréer un prototype de Sifteo 2.0 avec une autre technologie dans le but d'apprendre et de comprendre la programmation ? Pour répondre en partie à cette question, il a fallu d'abord appréhender le public cible. Nous devions déterminer les mécanismes d'apprentissage et les difficultés que peut rencontrer un étudiant lorsqu'il s'initie à la programmation. Cette étape de recherche pédagogique a permis d'évaluer l'utilité d'un jeu tangible pour aider l'étudiant à comprendre et appréhender les notions de fonction. Grâce à ces connaissances, nous avons imaginé un parcours pédagogique utilisant un jeu tangible sur base des fonctionnalités. Restait alors à examiner la possibilité de réaliser concrètement ce prototype.

Concernant l'aspect technique de la question, l'idée de départ était d'utiliser une Raspberry Pi Zero et un ePaper Papirus pour représenter le cube tangible. Il nous fallait alors explorer la possibilité de transposer les fonctionnalités clés du Sifteo cube. Une analyse a permis ainsi d'extraire les différentes fonctionnalités dont Sifteo dispose par rapport à ses concurrents.

Pour réaliser un cube tangible, nous avions besoin de plusieurs composants électroniques puisque la Raspberry ne disposait pas de toutes les technologies des Sifteo cubes. Nous devions déterminer le moyen utile de remplacer ces composants pour réaliser les fonctionnalités.

Nous avons réussi à intégrer le composant accéléromètre avec le ePaper PaPiRus, permettant ainsi de réaliser plusieurs fonctionnalités : l'inclinaison, le secouement, la simulation de la pression sur l'écran et l'affichage.

Ainsi pour répondre à la question de recherche, un prototype "Sifteo 2.0" est réalisable avec au moins trois composants externes à la Raspberry. Les deux premiers sont l'accéléromètre et le ePaper PaPirus. La fonctionnalité "contact entre cubes voisins" nécessitait un composant supplémentaire permettant de communiquer avec une autre Raspberry. Il reste à déterminer celui-ci. Il doit permettre de détecter le contact entre cubes voisins. Pour tester cette fonctionnalité, nous aurions dû posséder plusieurs dispositifs. Or, notre système est très peu maniable car les câbles de connexion sont externes au capteur. Tant que tout le dispositif ne sera pas intégré en un seul bloc, nous aurons des difficultés à avancer dans la recherche car les données ne sont pas précises.

Nous avons étudié la réalisation d'un cube. Dans des projets futurs, nous devrions nous focaliser sur plusieurs éléments importants. D'un point de vue étude, il est essentiel de mesurer l'impact de ce projet. Plusieurs études à long terme doivent être menées sur un échantillon d'étudiants afin d'avoir la certitude que ce type de solution les aident réellement dans l'apprentissage. D'un point de vue purement technique, nous devons nous focaliser sur l'intégration de plusieurs nano-ordinateurs et vérifier s'ils sont capables de produire un jeu adéquat. D'autres tests techniques doivent être effectués, comme par exemple, le traitement des données de l'accéléromètre. Ils permettront d'extraire des résultats et ainsi de calibrer les mouvements du cube en fonction de sa position dans l'espace.

Annexes

Annexe 1

1 Utilisation Raspberry Pi Zero [1]

Utilisation Raspberry Pi zero et PaPiRus pour simuler un sifteo.

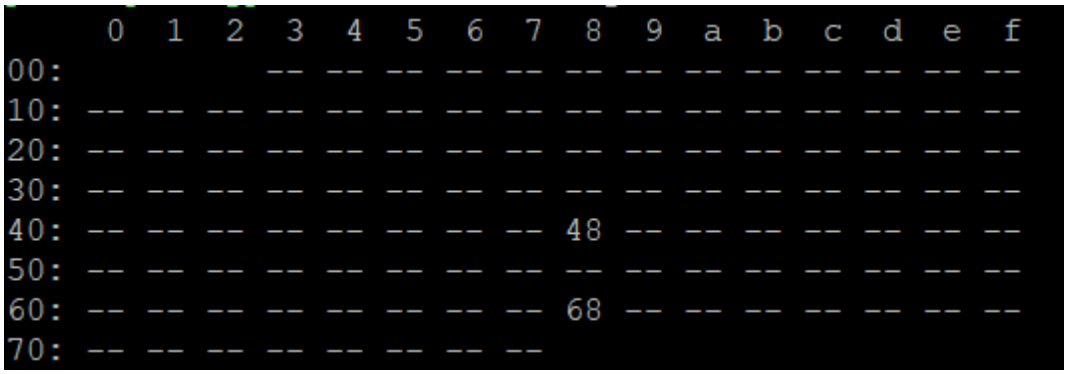

FIGURE 8.1 – Connexion PaPiRus et Accéléromètre sur Raspberry Pi Zero

— 48 : ePaper PaPiRus
— 68 : Accélérometer - MPU 9250

1.1 Quelques commandes clés

Ce sont les commandes les plus utilisées afin de vérifier s'il n'y avait pas de faux contact lié aux pins.

1. **Commande sous Raspberry** :
   — *i2cdetect -y 1* : savoir si les composants sont bien connectés à la raspberry ;
2. **Commandes sous PaPiRus** :
   — *papirus-test* savoir si l'écran est endommagé. Elle indique si l'écran est fonctionnel ou non ;
   — *papirus-write "Coucou"* permet d'écrire un message sur le ePaper ;

2 Images

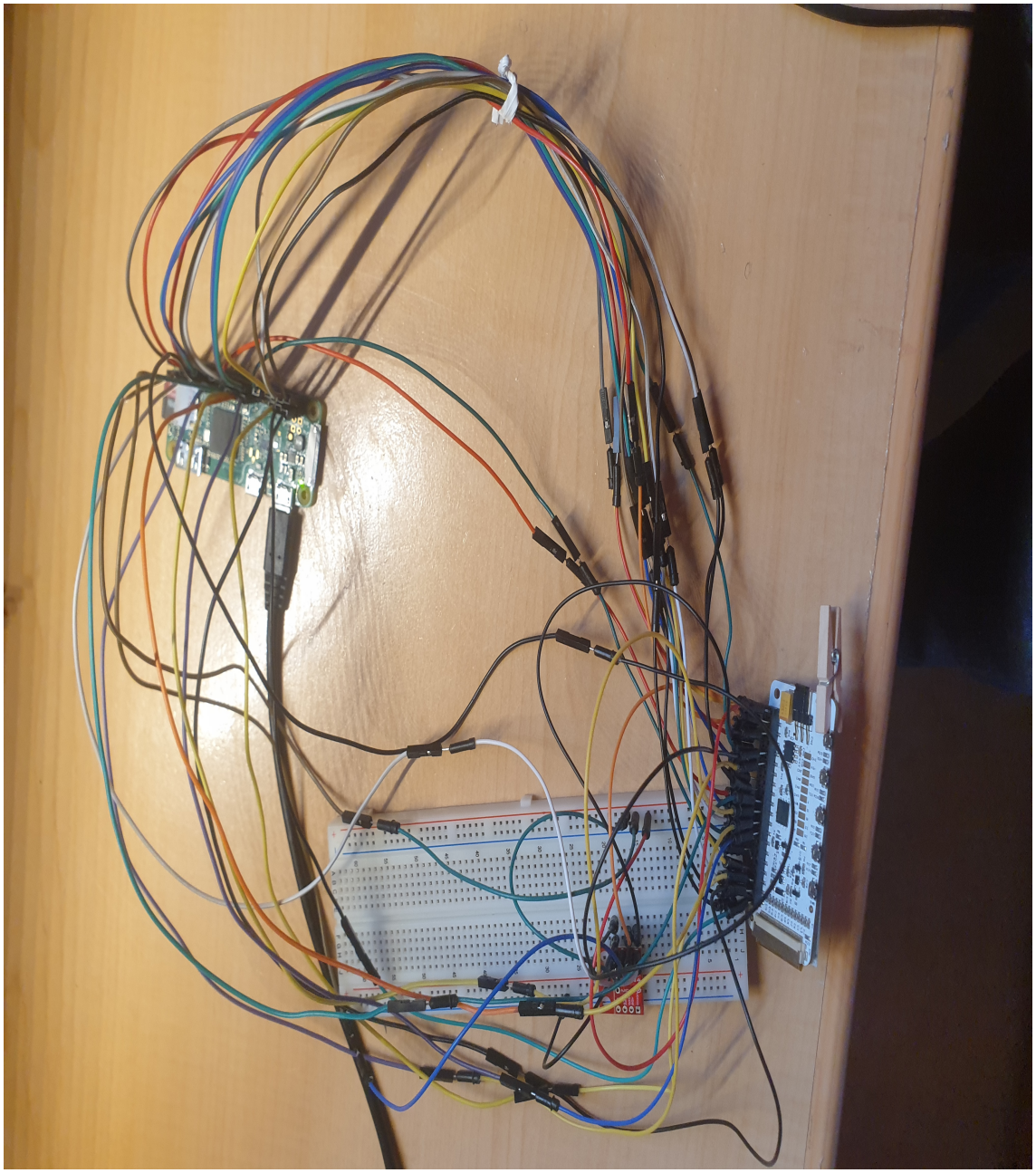

FIGURE 8.2 – Connexion réelle PaPiRus et Accéléromètre sur Raspberry Pi Zero

3 Tableau

Voici les différentes données obtenues par le capteur MPU-9250. Ces données correspondent aux 3 axes (x, y z) de l'accéléromètre.

Point neutre		
Axe X	Axe Y	Axe Z
0.075	-0.014	-1.038
0.076	-0.013	-1.039
0.075	-0.014	-1.038
0.075	-0.01	-1.04
0.075	-0.011	-1.037
0.077	-0.014	-1.043
0.073	-0.014	-1.04
0.074	-0.011	-1.038
0.076	-0.012	-1.045
0.075	-0.011	-1.041
0.075	-0.012	-1.041
0.073	-0.012	-1.04

Coordonnées lorsqu'on l'incline vers la droite à 90°		
Axe X	Axe Y	Axe Z
0.036	-1	-0.004
0.039	-0.994	0.072
0.043	-0.991	0.09
0.042	-0.995	0.079
0.041	-0.994	0.084
0.037	-0.992	0.075
0.043	-0.992	0.107
0.043	-0.991	0.106

Coordonnées lorsqu'on l'incline vers la gauche à 90°		
Axe X	Axe Y	Axe Z
-0.022	0.947	-0.36
-0.02	0.945	-0.355
-0.02	0.947	-0.353
-0.022	0.946	-0.355
-0.021	0.951	-0.356
-0.02	0.95	-0.359
-0.02	0.951	-0.36
-0.027	0.956	-0.332

Coordonnées lorsqu'on l'incline vers l'arrière à 90°		
Axe X	Axe Y	Axe Z
-0.959	-0.005	0.035
-0.958	-0.008	0.015
-0.958	-0.005	0.06
-0.956	-0.007	0.034
-0.96	-0.007	0.025
-0.956	-0.006	0.027
-0.958	-0.007	0.048
-0.957	-0.007	0.052

Coordonnées lorsqu'on l'incline vers l'avant à 90°		
Axe x	Axe Y	Axe Z
1.031	-0.032	-0.212
1.029	-0.037	-0.196
1.031	-0.032	-0.198
1.03	-0.034	-0.193
1.037	-0.028	-0.2
1.036	-0.029	-0.194
1.033	-0.031	-0.195
1.027	-0.036	-0.19

Voici les coordonnées lorsque l'accéléromètre est secoué de gauche à droite au niveau du table.

l'accéléromètre est secoué de droite à gauche		
Axe X	Axe Y	Axe Z
-0.042	-0.037	-1.018
0.127	0.583	-0.974
-0.149	-0.161	-1.017
0.009	0.12	-1.025
0.022	0.311	-0.997
-0.176	-0.399	-0.936
0.109	0.578	-0.97
-0.209	-0.494	-1.013
-0.034	0.041	-1.073
-0.081	-0.097	-1.051
-0.136	-0.199	-0.96
-0.215	-0.376	-0.862
-0.077	-0.022	-1.032

Table des figures

Bibliographie

[1] Richard Grimmett. *Getting started with Raspberry Pi Zero*. PACKT, 2016.

[2] Claire Chartier et Matthieu Scherrer. L'intelligence artificielle existera si nous renonçons à utiliser la nôtre. `https://www.lexpress.fr/actualite/sciences/l-intelligence-artificielle-existera-si-nous-renoncons-a-utiliser-la-notre_2072247.html`, 2019.

[3] Bodart Antoine. L'utilisation des interfaces tangibles pour l'apprentissage des concepts de programmation chez les jeunes. Master's thesis, Université de Namur, 2017-2018.

[4] Hiroshi Ishii and Brygg Ullmer. Tangible bits : towards seamless interfaces between people, bits and atoms. In *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, pages 234–241. ACM, 1997.

[5] David Merrill, Emily Sun, and Jeevan Kalanithi. Sifteo cubes. In *CHI'12 Extended Abstracts on Human Factors in Computing Systems*, pages 1015–1018. ACM, 2012.

[6] David Merrill, Jeevan Kalanithi, and Pattie Maes. Siftables : towards sensor network user interfaces. In *Proceedings of the 1st international conference on Tangible and embedded interaction*, pages 75–78. ACM, 2007.

[7] Fred. 3d robotics rachète sifteo. `https://www.helicomicro.com/2014/07/31/3d-robotics-rachete-sifteo/`.

[8] Jennifer Bergen. Sifteo intelligent cubes use nfc to interact with each other. `https://www.geek.com/games/sifteo-intelligent-cubes-use-nfc-to-interact-with-each-other-1411787/`, 2011.

[9] Jérôme Delécraz. La théorie de piaget : les stades du développement cognitif de l'enfant, est-ce que votre enfant se développe selon son âge ? `https://blog.cognifit.com/fr/theorie-de-piaget/`, 2018. Consulté le 1 février 2020.

[10] Zahid Ullah, Adidah Lajis, Mona Jamjoom, Abdulrahman H Altalhi, Jalal Shah, and Farrukh Saleem. A rule-based method for cognitive competency assessment in computer programming using bloom's taxonomy. *IEEE Access*, 7 :64663–64675, 2019.

[11] Patricia Lamas. Taxonomie de bloom : domaine cognitif. `https://patricialamas.net/taxonomie-de-bloom-domaine-cognitif`, 2020. Consulté le 1 mai 2020.

[12] Benoit Frenay. Introduction à la programmation, 2019.

[13] Luc Geurts, Vero Vanden Abeele, Kevin Van Keer, and Ruben Isenborghs. Playfully learning visual perspective taking skills with sifteo cubes. In *Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play*, pages 107–113. ACM, 2014.

[14] Clément Pillias, Raphaël Robert-Bouchard, and Guillaume Levieux. Designing tangible video games : lessons learned from the sifteo cubes. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 3163–3166. ACM, 2014.

[15] Florian Scharf, Thomas Winkler, Claudia Hahn, Christian Wolters, and Michael Herczeg. Tangicons 3.0 : an educational non-competitive collaborative game. In *Proceedings of the 11th International Conference on Interaction Design and Children*, pages 144–151. ACM, 2012.

[16] Seth Hunter, Jeevan Kalanithi, and David Merrill. Make a riddle and telestory : designing children's applications for the siftables platform. In *Proceedings of the 9th International Conference on Interaction Design and Children*, pages 206–209. ACM, 2010.

[17] Pejman Sajjadi, Edgar Omar Cebolledo Gutierrez, Sandra Trullemans, and Olga De Troyer. Maze commander : a collaborative asynchronous game using the oculus rift & the sifteo cubes. In *Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play*, pages 227–236. ACM, 2014.

[18] Mark Goadrich. Incorporating tangible computing devices into cs1. *Journal of Computing Sciences in Colleges*, 29(5) :23–31, 2014.

[19] José María Rodríguez Corral, Antón Civit Balcells, Arturo Morgado Estévez, Gabriel Jiménez Moreno, and María José Ferreiro Ramos. A game-based approach to the teaching of object-oriented programming languages. *Computers & Education*, 73 :83–92, 2014.

[20] Silvia Gabrielli, Rosa Maimone, Cristina Costa, Antonio Ascolese, Johanna Jonsdottir, Wolfhard Klein, and Gabriel Bendersky. A game-based solution for in-home rehabilitation. In *International Internet of Things Summit*, pages 112–117. Springer, 2014.

[21] Thuong N Hoang, Deborah A Foloppe, and Paul Richard. Tangible virtual kitchen for the rehabilitation of alzheimer's patients. In *2015 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 161–162. IEEE, 2015.

[22] Marijke Vandermaesen, Tom De Weyer, Kris Luyten, and Karin Coninx. Physicube : providing tangible interaction in a pervasive upper-limb rehabilitation system. In *Proceedings of the 8th International Conference on Tangible, Embedded and Embodied Interaction*, pages 85–92. ACM, 2014.

[23] Monika D Heller, Kurt Roots, Sanjana Srivastava, Jennifer Schumann, Jaideep Srivastava, and T Sigi Hale. A machine learning-based analysis of game data for attention deficit hyperactivity disorder assessment. *GAMES FOR HEALTH : Research, Development, and Clinical Applications*, 2(5) :291–298, 2013.

[24] Chris Wiltz. Cogcubed diagnoses adhd with video games. `https://www.mddionline.com/cogcubed-diagnoses-adhd-video-games`, 2013.

[25] Ibrahim Said Ahmad, Saminu Muhammad Aliyu, and Hafsa Kabir Ahmad. Towards the engagement of children with adhd using sifteo cube technology. *Asia-Pacific Journal of Information Technology and Multimedia*, 8(2), 2019.

[26] Ricardo Langner, Anton Augsburg, and Raimund Dachselt. Cubequery : Tangible interface for creating and manipulating database queries. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces*, pages 423–426. ACM, 2014.

[27] Yves Rangoni, Valérie Maquil, Eric Tobias, and Eric Ras. Implementing widgets using sifteo cubes for visual modelling on tangible user interfaces. In *Proceedings of the 2014 ACM SIGCHI symposium on Engineering interactive computing systems*, pages 205–210. ACM, 2014.

[28] Rafael Alves Roberto, Daniel Queiroz de Freitas, Francisco Paulo Magalhaes Simões, and Veronica Teichrieb. A dynamic blocks platform based on projective augmented reality and tangible interfaces for educational activities. In *2013 XV Symposium on Virtual and Augmented Reality*, pages 1–9. IEEE, 2013.

[29] Bert Schiettecatte and Jean Vanderdonckt. Audiocubes : a distributed cube tangible interface based on interaction range for sound design. In *Proceedings of the 2nd international conference on Tangible and embedded interaction*, pages 3–10, 2008.

[30] Wooi Boon Goh, LL Kasun, Jacquelyn Tan, Wei Shou, et al. The i-cube : design considerations for block-based digital manipulatives and their applications. In *Proceedings of the Designing Interactive Systems Conference*, pages 398–407. ACM, 2012.

[31] `https://www.raspberrypi.org/`.

[32] Pi Supply. Papirus - the epaper screen hat for your raspberry pi. `https://www.kickstarter.com/projects/pisupply/papirus-the-epaper-screen-hat-for-your-raspberry-p`, year = oct 2018.

[33] Reginald Watson. How to interface an imu sensor with raspberry pi. `https://maker.pro/raspberry-pi/tutorial/how-to-interface-an-imu-sensor-with-a-raspberry-pi`, 2019.

[34] Vincent Kerhoas. Présentation du capteur mpu9250. `https://www.enib.fr/~kerhoas/rescapt_cours_mpu9250.html`.

[35] F. Ferroro. Les accéléromètres : Electronique pour arduino, 2017.

[36] Luc de Brabandère. *Petite philosophie de la transformation digitale*. Manitoba, Mars 2019.